JPRS-UCC-90-005
24 JULY 1990

FOREIGN
BROADCAST
INFORMATION
SERVICE

JPRS Report

Science & Technology

USSR: Computers

Science & Technology
USSR: Computers

JPRS-UCC-90-005 CONTENTS 24 July 1990

Computerization Proposal Criticized

907G0067A Moscow SOTSIALISTICHESKAYA INDUSTRIYA in Russian 19 Dec 89 p 2

[Article by Sergey Panasenko, "Mirages of Informatization"]

[Text] **"The leadership of no other country devotes as much attention to the development of informatization, as ours, yet there is not a single country where these efforts could have had such an insignificant effect." So began the draft concept for informatization of Soviet society, prepared almost a year ago by a group headed by Academician D. Gvishiani. In a summarized version of the draft, compiled from proposals by this group and two others—from the UkSSR Academy of Sciences Institute for Cybernetics imeni V.M. Glushkov and the VNII For Problems in Computer Hardware and Information Science—you will not find these words. This is nothing strange. After all, after such a statement we either had to cease multiplying fruitless "efforts," or create a document, radically different from those which the country had seen before. However, this did not happen.**

'Tomorrow at 7:00...'

All the discussions of whether or not we should be "informaticized," of who is "for" and who is "against" it, attest to the complete misunderstanding of that which has occurred. One may object to the coming of winter, but it nonetheless comes in its own time. An information society is an inevitable stage of civilization's development. The difference lies only in time. Some countries have already entered this stage today, others have come close to it, and a third face another decade of growth. However, for everyone, without exception, there are no alternatives to an information society.

This does not mean that we talk about an information society like a rendezvous "tomorrow at 7:00." We can slow down or accelerate our approach to it, facilitate or complicate it, but the capabilities of one person or a group of people, even of those invested with power, is highly limited in this area. Meanwhile, in the USSR, due to reasons that we will not discuss here, a view of history and its laws, as though of soft wax in the hands of a sculptor, has taken root. Here, a large number of intelligent and honest people sincerely assume that an information society can be successfully built like a shock construction site: pass a resolution, allocate funds, choose the workers—onward! The draft concept in question proceeds precisely from this.

Rigid, direct state control over the process of informatization presumes implementation by the old principle of "so much of something by such and such a time." All spheres of human activity, all points for application of efforts are listed, one after another, arranged according to order of ability to be informaticized.

Centralized outlays for informatization have been clearly named: approximately 720 billion rubles over the three coming 3-year periods, or an average of 48 billion per year! The growth rates for the production volumes of hardware and components for computer equipment in the 5-year periods are being assigned: 13 percent on the average. Yet these figures evoke no doubts from the authors of the concept. "Will grow... will achieve..."—for them, the question is answered. Without hesitation, an increase by a factor of 8-9 in the volumes of development of all software tools, and by a factor of 30—for basic software, is being planned. The country's need for computer hardware has been calculated right up to the year 2005: for supercomputers, work stations and personal computers. The reduction in cost for a unit of hardware has been calculated, the number of data base and data bank systems has been planned, and so on.

Try to question the sensibility of such an approach, or rebuke the authors for their love of the command-administrative style! You will hear that all countries have acted thus, "having given informatization the highest priority, having subordinated the basic resources and efforts to this goal."

It is indisputable that the world's developed countries have marked informatization number one on their lists of national programs and that the state watches over this process in all ways. However, our similarity to the rest of the world ends with recognition of this fact.

Everything, Like We Have?

By general admission, the most "informaticized" country in the world today is the U.S. Here, 426 of the 816 scientific and technical data banks and 719 of the 1,035 economic data banks existing in the world are concentrated. American companies and universities hold the basic patents in the field of information technologies and conduct the most important basic research. The parameters of American electronics hardware are becoming de facto world standards. However, little is known in the USSR about the process of informatization of American society: perhaps that is why the strategic computer initiative, announced 6 years ago, enjoys such fame.

Yet, the informatization of America was a typical "initiative from below." The National Commission on Libraries and Scientific Information put out the first programmatic document on informatization on its own initiative in 1975. A year later, the public organization "Internal Council for Protection of the Right to Privacy" sent the report, "National Information Policy," to the president. Finally, the American Association of Libraries presented a third document in 1978: "In the Information Age. Prospects for Nation-Wide Informatization Activities," which also set the foundation for all subsequent specific decisions by the government.

In practical work, business gets by without Washington's helping hand. As an example of self-organization, let me cite the Semiconductor Research Corporation [SRC] consortium, founded in 1982 for the coordination and implementation of research in semiconductor technology. Giants like AT&T, DuPont, General Electric, IBM, Westinghouse, and several dozen other leading companies are part of this consortium. In 1985 alone, the SRC handed out 40 different projects to 30 U.S. universities. The Microelectronics and Computer Technology Corporation is engaging in similar activity: it is yet another similar association, but from a somewhat different sphere. The role of the government in both cases is reduced to the regulation of anti-trust legislation, so that it does not obstruct the activity of companies.

The second important task of the federal government is to establish standards for information equipment purchased by the Pentagon and by other federal departments (note: for **that purchased by the government**, not for everything produced in the country!). By this token, a definite technological level, required in the country today from the viewpoint of legislators, is assigned.

One could object that the example of the United States is unsuitable for us: countries which are catching up are of more interest. Well then, let us consider Western Europe, where several national and extra-national programs for the development of informatization are operating: "EUREKA" and "ESPRIT." It is easy to see that these programs are nothing other than the government's answer to the SOS signal, given by the electronics industry of West European countries. "We are losing!" The industrialists are pestering legislators: "We know what we must do, but you should help us." The governments are not refusing.

In this regard, there are different tactical solutions. France, for example, after a program for development of information technologies passed in 1983, nationalized this sector of the economy. The state became owner of five industrial groups and 39 banks, related to the manufacturing of electronics and computer hardware, and to information technologies. Additional protectionist measures were passed, for example, on development of a communications protocol for the National Information-Computer Network, incompatible with the protocols of the IBM networks which have in fact become world standards.

Great Britain chose another path. Since experiments with nationalization of whole sectors there in 1982, when a national program appeared for research in information technologies, concluded deplorably (especially in the scientific sectors), the stakes were set on encouraging private initiative and competition. It is noteworthy that the developer of this program was the chairman of the board of the "British Telecom" Company, John Elvey, and one of the elements of the program was disenfranchisement of the monopoly by this company on internal communications networks in the country: the American ITT firm received a similar license. The state Center for Automated Design Systems was sold to a consortium of British companies, headed by International Computers Ltd. for 1.4 million dollars. The results of these and similar actions were quick to appear: in 1986 already, out of 33 prizes received by British companies for the best developments, 25 related to electronics and information science.

In West Germany, the state 4-year program for development of NIOKR in information technologies was passed in 1985. The positions of the German electronics and information industry in the world are more solid than their European competitors' and, possibly, it is because in the FRG there is not a single agency which coordinates research in this field: it is done independently by more than 30 private and state institutions. State participation is reduced only to a more or less strong financial assistance for competitive projects and exploratory work. The FRG government also does not take any protectionist steps whatsoever to guard its internal information technology market, believing competition the best incentive for development.

In short, despite all the differences in tactics, strategically the West European countries are operating identically: they are creating **conditions** which contribute to the development and strengthening of the corresponding sectors of the economy. However, not one government is directly guiding the industry or dictating how much and what kind of computers to produce and at what price to sell them.

There is no paradox whatsoever in the outwardly paradoxical quotation from D. Gvishiani's concept group, with which I began this article. The insignificant effect was predetermined precisely by the incommensurate interference of state agencies in informatization. After all, if one agrees with the viewpoint of the American researcher Anthony Ettinger, that "every society is an information society," since in one way or another it mandatorily produces and uses one kind of information or another, then informatization becomes a synonym for society's growth, an indicator of its degree of maturity. You can raise a child, but you cannot make him grow...

'You Can Lead a Horse to Water...

...but you cannot make him drink.' So goes the English saying. However, the draft concept is trying precisely to force us to "drink" informatization, not reckoning with thirst, i.e., the economy's capability of accepting informatization and its need for it.

No, the authors of the concept remind us that in our country "the socioeconomic conditions, which would predetermine the vital necessity of informatization and its results for mass use are insufficiently developed" and that "the existing economic mechanism does not contribute to broad-scale and effective informatization..." However, instead of analyzing the situation, instead of even an approximate prediction (after all, events may develop differently), the authors limit themselves to an incantation about the appearance "in connection with perestroyka and radical economic reform... of prerequisites, such that informatization will become a necessary condition for society's survival." What these "necessary conditions" are is explained by the chapter, "Goals of

Informatization," where the first point in the section on economics reads: "We must bring the volumes and structure of produced output into accordance with the scientifically substantiated needs of society by utilizing modern methods of management, accounting, analysis, and forecasting on the basis of introducing new information technologies and decision-making systems."

Everyone who remembers the epic with ASU at a cost of 120 billion rubles will notice something familiar about these lines. Of course, at that time they promised society to bring order into the chaos of our economy "by utilization of contemporary methods of management, accounting, analysis, and forecasting." It did not succeed, because the roots of chaos existed neither in accounting, nor in analysis. Even now, they do not lie there. Why the certainty that the desired result will be achieved this time?

In asserting that informatization is an inevitable stage of civilization's development, I have somewhat simplified the problem. An information society is logically inevitable only for countries, whose economy is developed intensively. For extensive economies, and for the time being ours is such, informatization does not follow from progress, albeit because progress here does not occur in depth, but in width. If something pushes the leaders of such economic systems toward informatization, it is only a depressing comparison of the standards of living and decline of the country's prestige and importance in the world arena, and there is still an inspired belief that informatization is the genie in a bottle from the "*Arabian Knights*", which we must indifferently rub along the shore of some body of water: the Atlantic Ocean, the Yellow Sea, or Chukhlomskiy Lake.

In this regard, the explanation is: We have no choice, either informaticize ourselves as soon as possible, no matter what, or, as written in the draft, "in order to overcome serious negative phenomena in the country's economy, the investment of additional resources, ensuring the re-equipment of basic inventories of microelectronics and computer hardware and other information science systems, will be required." We again return to the problem of choice. Meanwhile, let us ask ourselves two questions: Is the country in a condition to invest these "significant resources," and does it have a need for this today?

These resources, as we recall, are colossal: 48 billion on the average annually. For comparison: all centralized capital investments are planned to amount to 83.5 billion in 1990. Moreover, these expenses include only the outlays for computer hardware and information science. However, computer users here continually complain about the "jumping" of line voltage: money is also needed to eliminate this problem, as well as to produce a sufficient number of false floors for computer centers or printer ribbons... It will become necessary to build: this requires cement, steel, wood, and transportation. The authors of the concept have overlooked the funds needed for these purposes.

Yu. Lapshin, head of the information science department of the Academy of the National Economy, spoke at a meeting of the Commission of the Union Council on Problems of Transportation, Communications, and Information Science in November 1989. "You ask for money," he asks the project authors, "but what do you promise in exchange? Percentages of growth of labor productivity again? Then Gosplan should take into account, in its calculations, the reduction in the production of machine tools or other means of production and the decrease in capital investments. However, there are no such computations in your concept...

People's Deputy G. Anisimova also spoke of the poor resource development of the concept at this meeting. However, I think, she would have been surprised if she had found out that the planning bodies... had almost nothing to do with the draft concept! In the words of G. Mirskiy, head of the Gosplan subdepartment for automation equipment, Gosplan did not visa the figures used in the draft concept, and therefore they must be considered only the wish of the compilers of the project, and by no means as a possibility. Yet, after all, if the deputies accept the concept, the government will have to develop a specific program for informatization of the country on the basis of these figures!

The demand for informatization, the economy's real need for networks, data banks, computer and telecommunications equipment is the second question that we must answer. Most often, the incredible sums which enterprises will pay vendors for various kinds of computer equipment are referred to to corroborate such demand. This argument was also heard at the above-mentioned commission conference. I question that the 5-figure sums, paid by enterprises from the development fund (i.e., from money that is "dead" for the collective under the current economic situation, part of which moreover they are promising to withdraw into the budget) for several types of imported computer equipment, attest to their interest in informatization. This is only an interest in acquisition of a set of good equipment, which may or may not prove useful, but which, like the canvases of the masters, will grow valuable in the future. Interest in informatization, demand for it, is primarily a demand for information. Yet, for the time being this is not evident.

As a result of all the latest extraordinary measures by the government, we are further from a market, perhaps, than a year ago. Yet, only it would be able to stimulate interest in economic, scientific, and commercial information. Approximately 90 percent of machine building output is made at only one enterprise. Neither the monopolist producer, nor the doomed customer needs any kind of serious information.

Yet another typical feature is the wholesale and retail sale of computer centers, which is easily ascertained, having opened any issue of the capital's "Advertising Enclosure." At the same time, there is a sharp drop in demand for new information equipment: The Vilnyus

"Sigma" Production Association cannot collect enough orders for its fairly good SM 1700M ARM. Yet another figure, quoted at the recent seminar "Informatization of Soviet Society," by V. Nechiporenko, chief of the GKNT Joint Department for Information: on the initiative of labor collectives in Moldavia, 15 percent of low-level information services have already been shut down. There are no grounds to hope that the picture is fundamentally different in other regions.

Pluralism Without Choices

There is a great deal that I do not understand in this concept. I do not understand why one page justly states that "our country is faced with the task of determining its place and methods for participating in the international division of labor in the field of information science," while another lists the devices and technologies which "should be developed and mastered in series production," a list which repeats everything already known in the world of information equipment, from trivial telephones to exotic optical disks. I do not understand why in one place they suggest that "achievement by 1995 of a world technical and economic level and quality of leading domestic computer hardware and componentry is necessary, and by the year 2000—series and mass production," but in another, the position changes to the opposite: "We must reject the traditional directive of catching up with developed capitalist countries..." I do not understand how it is possible, on the threshold of a law-governed state, to devote only a few words to protecting the individual from information terror. Western experts still see a danger in informatization of increased control over citizens, manipulation of the mass information media, and intrusion in private life, and consider this almost the key question. In France, for example, a law was passed in 1978 on information science, card indexes, and freedom, which was proposed by the National Commission on Information Science and Freedoms. The year before, a similar law was passed in the FRG. However, there is not a word about this in the concept.

I do not understand the main point: Why should I consider the proposed variant for informatization of society the only one possible? Where is there even a mention of other possibilities, of other models? I clearly see the entire complexity of "socioeconomic relations" in the country, but why should I believe that "a radical change of these relations... cannot successfully be developed without solving the problem of informatization" in the proposed system?

In writing this article, I talked to a fairly large number of specialists in the field of computer hardware and information science, who did not participate in development of the draft concept. Many had not read this document and were not very eager to do so, not trusting it from the start. A frank skepticism prevailed among the rest. Professor E. Rakitov stated that he does not consider the compilation of such drafts productive and does not see a need for talking about the subject for long. V. Ivannikov, USSR Academy of Sciences corresponding member, expressed doubt that this concept is capable of saving the country. "We need a clear and simple plan of actions, an algorithm," he said. "Although I greatly doubt that, in this situation into which we have driven ourselves, anyone is in a state to propose such a program!" His titled colleague, who asked, however, not to be named, said the following: "The course of informatization is not determined by our desires and concepts, but by the level of technical and intellectual readiness of society. Concepts such as this simply are unnecessary." I could quote many similar such statements.

It is becoming clear from these talks that the main problem with the concept is not even the dubious nature of resource defrayment, but the plainly obvious plan for a "big jump." Ascertaining the sad state of the economy turns out to be not a starting point for forward movement, but a trampoline for this jump. Believing that the solution of today's problem lies in universal informatization of the national economy, they in fact are suggesting that the country endure the treatment of a disease for 15 years, in order, in a wonderful, informaticized future world, to finally put an end to present-day troubles.

However, specialists in industrial automation and robotization know that modern technologies give a substantial effect only where simpler, traditional methods for raising labor productivity have been exhausted. Robotization of chaos gives only a robotized chaos, not order. The same holds true for informatization.

It easily brings savings, when it is placed, like a brick on a brick, on top of previous technological solutions. However, what kind of bricklayer in his right mind would try to lay the second floor of a building on an unstable first, even if the neighboring building has been constructed up to the third floor already?

Are all traditional methods for gathering, processing, and disseminating information exhausted? Of course not. Otherwise, the deputy director of the State Public Scientific and Technical Library, the main scientific and technical library for the country, would not have uttered the bitter phrase: "Slogans about scientific and technical progress are not corroborated by the state's attitude toward scientific and technical libraries." Let me add: toward others as well. The fantastic congestion of inventories and the poverty of library workers has forced the GPNTB to take an unprecedented step: the library is selling its inventory! In my view, this can be considered nothing but a disgrace.

We immediately started out big, with informatization. Yet, many countries made a more modest first step: first, a program for computerization. So started South Korea and Brazil, and even the highly developed France began with a 1978 report on national policy on computerization. We go by without a report. Therefore, our computerization is not similar to any other in the world. All developing countries to the utmost encouraged the import of computers into the country—we impose unthinkable duties on them and on peripheral equipment. All countries are beginning with the simplest "turnkey" assemblies of reliable Western models, but we are building monster factories for series production of domestic personal computers, which are already yesterday's IBM computer. In our automatic telephone exchanges, 11 million numbers, i.e., approximately 30 percent, have outlived their service life and produce jibberish even in "telephone" mode, and they simply cannot handle "computer" mode—they must be replaced. Finally, the telephone network of Moscow—the capital (!)—was designed on the estimate that each telephone will operate on an average of 9 minutes per hour. That is, the appearance of even a slightly noticeable number of modems, devices for connecting personal computers to a telephone network, will cause a new wave of break-downs in the city.

Without the solution of these and other closely related problems, it is unthinkable to talk about the technical aspect of informatization.

Blind Informatization

One of the members of the Commission on Information Science, a people's deputy, told me that they had been asked not to give the text of the draft concept to journalists: the journalists will exclaim, they said, and you should not thus consider... Laudable modesty. Perhaps, I am really writing all this in vain? Let the deputies well consider, weigh, and evaluate everything...

I am very much afraid that nothing will come of this. Not because our deputies are such bad ones. They are good. However, there is a limit to human knowledge, a lower boundary, below which one looses the ability to skillfully judge. When there is no information, a decision is made carelessly or blindly.

Two books lie on the table in front of me. The first 165-page volume is called "*Computerized National Information Systems. Technological and Social Aspects.*" It was published in 1981 by the Administration for Technology Assessment, a special auxiliary branch of the U.S. Congress whose task is to help congressmen evaluate one or another technological innovation. The book contains a full review of what computerized information networks, which sprang up very actively in the mid-1980s in the U.S., entail. Described in simple and clear language, supplied with a number of diagrams and illustrations, it considers networks from all conceivable angles: structures, influence on employment, individual rights, public security, legal... If you have any doubts whatsoever after reading it, the administration will provide any additional explanations, so that before the issue is heard in the House of Representatives or the Senate, the congressmen will possess exhaustive information. This, you will note, is in the most informaticized society on the planet, in a parliament where almost everyone has a higher university education.

The second 320-page book is called "*Concept for Development of Information Engineering.*" It was published this August jointly by the FRG Ministry of Research and Technology and the FRG Ministry of Industry. I cannot think of anything related to the informatization problem, even in part, that was not covered in this work. Once again—simple, clear language, phrases in which one need not dig for the meaning, although the subject is not among the easiest.

Then, I pick up a 40-page brochure—our domestic version of the concept—and feel pained. Pained for those who compiled this tongue-twister, not even trying to make it understandable for the uninitiated. Pained for those who, relying on this "short course," will have to hurriedly resolve a most complex problem. Very well: our committees are still just learning how to work with the parliament. However, why is the parliament not learning from this? After all, I can guarantee you that half of the members of the Commission on Questions of Transportation, Communications, and Information Science simply did not understand what this thin booklet was talking about. How come they did not say: Dear comrades, write your concept so that we are not confused by terms and so that we know what we are voting for?

However, the telegraphic style is convenient for the authors. Thus, they can limited themselves to postulates, the detailed development of which might unexpectedly reveal the weakness of their arguments. Thus informatization is successfully "declared," having forgotten the eloquent phrase: "It does not take long to declare a republic, but where do you get republicans?"

In April 1989, an international conference was held in Suzdal on the informatization of Soviet society, the main event of which was the presentation of three initial concepts for the informatization of the country. On the final day of the conference, I became interested by the opinion of its scientific secretary, E. Bernshteyn, author of his own concept, which is well-known among information workers.

"There is no concept," he answered despondently. "However, they will pass it, allocate funds, and start to put it into practice. Collapse will follow. When society, God willing, succeeds in realizing the importance of the problem, it will be ready for it. Before that time, everything is in vain."

Will his prediction come true? For the time being, it is...

UDC 681.324

On One Approach to the Organization of the Access of Users to Information of Applied Tasks Which Function at Computer Centers

907G0095 Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 90 (manuscript received 22 Nov 88) pp 53-57

[Article by S. V. Balandina and V. M. Spasibukhov]

[Text] Insofar as the need for controlled collective use of programs and data at computer centers (VTs) is increasing, the need for systems which implement the principles of protection is becoming far more urgent [1-3].

Therefore, in addition to the "Planning and Accounting of Computer Jobs," "Organization of the Computing

Process," "Monitoring of Operation," "Reliability," and "Formation, Reception, and Transmission of Data" subsystems, a "Protection of Information Against Unauthorized Access" subsystem was included among the functional subsystems of large computer centers that are being developed in the Department of Automated Control Systems of the Azerbaijan SSR Academy of Sciences.

Within this subsystem the following software modules are being elaborated: "Login," "User Passwords," "Processing of Query-Orders," and "Analysis" (see figure).

The "Login" software module is designed to verify the authorization of the requested connection between the terminals of users and computer centers and includes the tasks of assigning users to terminals; establishing a list of terminals which are served by the computer center during a given time interval; identifying and registering the terminal; monitoring and registering the authorization of the work of the user with the terminal; and monitoring and registering the task being requested.

The "User Passwords" module facilitates the assignment, maintenance, and accounting of computer center user passwords in order to prevent unauthorized access to protected information of the tasks executed at the computer center.

User access to information of the applied tasks which operate at computer centers is regulated within this subsystem.

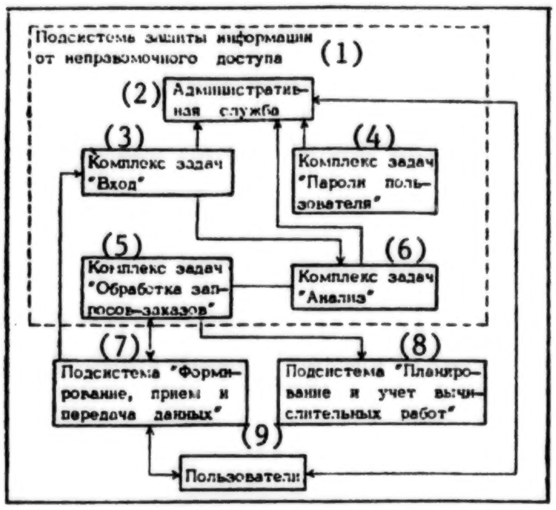

Подсистемь защиты информации от неправомочного доступа
(1)
(2)
Административная служба
(3)
Комплекс задач "Вход"
(4)
Комплекс задач "Пароли пользователя"
(5)
Комплекс задач "Обработка запросов-заказов"
(6)
Комплекс задач "Анализ"
(7)
Подсистема "Формирование, прием и передача данных"
(8)
Подсистема "Планирование и учет вычислительных работ"
(9)
Пользователи

Diagram of the Interconnection of Software Modules within the "Protection of Information Against Unauthorized Access" Subsystem

Key:—1. Subsystem of the protection of information against unauthorized access—2. Administrative service—3. "Login" module—4. "User Passwords" module—5. "Processing of Query-Orders" module—6. "Analysis" module—7. "Formation, Reception, and Transmission of Data" Subsystem—8. "Planning and Accounting of Computer Jobs" Subsystem—9. Users

Formalization of user queries makes it possible to plan, manage, and monitor the computer center information and computing resources that are necessary to fulfill the queries. Moreover, determining the structure and composition of the information of the query makes it possible to determine the necessary minimum amount of data that should come directly from the user himself, as well as the necessary amount of systems and reference data. This ensures both the processing of the query itself and the organization of the corresponding computing process for its execution. Here, restriction of user access both to the data directly requested and to the necessary application program processes should be determined.

And, finally, formalization of user queries is necessary and should facilitate accounting for computing resource expenditures, which are subsequently presented for payment.

Taking into account that in the future large computer centers will be united into a network, an analysis of the possible composition of computer center users, as well as their possible demands for information and computing services was made, in order to determine the structure and content of queries.

In this way the advisability of grouping computer center users according to the following classes was established:

—directive (managerial) users (directive agencies, individual ministries and agencies, as well as the corresponding officials);

—outside users (computer centers of ministries, departments, enterprises, and organizations, as well as individual officials of ministries, agencies, enterprises, and organizations);

—system users (the administration, developers of tasks, maintenance personnel, programmers of computer centers);

—individual users (people who in the future will acquire the necessary set of terminal equipment).

These classes of users are distinguished by the composition of the demands on the possible jobs and modes of operation at the computer centers, the group of tasks being accomplished, the periodicity and predictability of the accesses to data and programs, the regulations on the issuing of responses, the number of indicators which identify the user, and a number of other factors.

For example, a query for the accomplishment of a task is characteristic of directive users, but for them there is no need to prepare computer media and allocate machine time for debugging software, yet all these operations are characteristic of system users.

As a whole, the analysis of the classes of users, their possible queries, and the demands on the implementation of the computing process at computer centers for execution of these queries made it possible to begin developing standard structures (forms) for the queries of users at computer centers, which are called query-orders. Query-orders are divided into three types: accomplishment of tasks, allocation of machine time, and work on the preparation of data on computer media.

The structure of query-orders using Backus notation [4] has the following form:

< query-order > ::= < header > < body of order > < header > ::= < code of template > < header 1> divides < header 2 > < header 1 > ::= < date of request > < number of lines in template > < access restriction flag > < header 2 > ::= < date of transmission of query > < time of transmission of query > < number of lines in template > < access restriction flag > < body of order > ::= < user > < body 1 > divides < body 2 > divides < body 3 > < user > ::= < user 1 > divides < user 2 > divides < user 3 > < user 1 > ::= < code of user > < password > < user 2 > ::= < code of organization > < code of position > < password > < user 3 > ::= < code of organization > < code of subdivision > < code of position > < password > < body 1 > ::= < code of task > < code of type of settlements > < code of form of documents > < as of... > < number of copies > < time > < body 2 > ::= < code of task > < code of type of operations in system > < code of type of settlements > < code of form of document > < number of copies > < time > < body 3 > ::= < code of task > < code of type of operations in system > < code of type of medium of input information > < code of form of document > < registration number of volume > < code of flag of restriction of input information > < code of type of medium, on which information is carried > < code of type of operations on preparation of data > < amount of data > < registration number of volume, on which information is carried > < number of copies > < number of file on copy > < time > < time > ::= < date of completion of work > < time of completion of work >

As a whole the query-orders submitted by users to computer centers are drawn up, as a rule, for industrial functional tasks (in case it is later necessary to make changes in deadlines for tasks and the composition of the documents to be issued); for the allocation of computer center computing resources for performing debugging operations on the programming and pilot operation of tasks; and for work on the preparation of data on machine media. If the agreed- upon composition of the documents to be issued and the deadline for work on industrial functional tasks suit the user, query-orders are not submitted. These data are entered in the systems tables and serve as the basis for planning and management of the computing process of the computer center, and are adjusted as necessary.

Thus, the use of query-orders makes it possible to standardize the access of computer center users and ensures the organization of all the subsequent stages of the computing process: access proper, monitoring, planning, and management.

The development of the special "Processing of Query-Orders" module for the "Protection of Information Against Unauthorized Access" functional subsystem (see the figure) is envisaged for processing query-orders. The purpose of this module is to verify the correctness of query-orders, to prevent unauthorized use of computer centers and user budget overruns, and to prepare information for planning computing operations at computer centers and analysis of violations of query-order submission procedures.

Registration in the appropriate systems logs is envisaged with respect to all rejected access attempts. Moreover, if a user exceeds a specific number of incorrectly input passwords during the day, the following information is output to the display screen of the authorized protection agent: the user identifier, the list of incorrectly input passwords, and the time and number of the terminal from which they were input. Accordingly, special systems logs are being introduced [1-3]. They record attempts at matching passwords and the terminals from which the user incorrectly identified himself, as well as attempts by users who correctly name the password, but are not found among the other identifying data user data in what is called the user authorization profile.

Thus, the systems logs, which are generated by the "Login" and "Processing of Query-Orders" modules, serve as input information for the "Analysis" subsystem. The purpose of this subsystem is to analyze the causes of unauthorized access at computer centers and to identify and even forecast possible attempts at unauthorized access.

Using the "Analysis" module (see the figure), the authorized protection agent can periodically call to the display screen or print out these logs, sorted by different fields, which will make it possible to determine the user authorization profiles and the denials of access by terminal, user, and task.

If, in the course of processing user query-orders for information and computing services, no errors or violations are detected, the file of received query-orders is sorted for subsequent processing in the "Planning and Accounting of Computer Jobs" functional subsystem. It is processed using the following time intervals: for the current day to 1600 hours, for the current day after 1600 hours, and for subsequent days.

The information on accepted or "rejected" tasks is accordingly reported to the user with a diagnosis of the errors detected in the submitted query-order. The rejection codes used are presented below (the number of the digits is designated by the symbol X): *Structure of Overall Query-Order Rejection Codes:*
—the budget has been exceeded (X);
—the account has not been opened (X);
—the number of rejected tasks (XX);
—the number of errors in the header of the query-order (XX). *Structure of Query-Order Header Rejection Codes:*
—the time of the query was incorrectly indicated (X);

—the day of the query was incorrectly indicated (X);
—the password was not found in the profile (X);
—the position was not found in the profile (X);
—the subdivision was not found in the profile (X);
—the organization was not found in the profile (X);
—the position was not found in the classifier of positions (X);
—the subdivision was incorrectly indicated (X);
—the user was not found in the classifier (X);
—the user group was incorrectly indicated (X);
—the user type was incorrectly indicated (X);
—the template code was incorrectly indicated (X). *Structure of Task Rejection Codes:*
—the calculation type was not found in the classifier (X);
—the type of operations in the system was not found in the classifier (X);
—the task is not in the profile (X);
—the user does not have access (X);
—the password was incorrectly indicated (X);
—the password is missing in the query-order (X);
—the user was not found in the profile (X);
—the ordinal number of the rejected task (XX);
—the task was not found in the classifier of tasks (X). *Structure of Document Rejection Codes:*
—the hour of document issuance was incorrectly indicated (X);
—the minutes document issuance were incorrectly indicated (X);
—the seconds document issuance were incorrectly indicated (X);
—the date of document issuance was incorrectly indicated (X);
—the date indicated in the query-order is not in the current month (X);
—the month was incorrectly indicated (X);
—the year was incorrectly indicated (X);
—the number of copies was incorrectly indicated (X);
—the document is not in the profile (X);
—the access of the user does not correspond to the restriction flag of the document (X);
—the document was not found in the classifier of tasks (X).

It should be noted that a special information and reference file database was developed for the "Processing of Query-Orders" module. The processing of query-orders at computer centers on the basis of the information and reference database is quite simple owing to the standardization of query-orders, the number of control checks is stabilized, and the proportion of special control procedures that implement more complex checks than the logical relationships check decreases.

Another unquestionable advantage of using the information and reference database is the fact that the user is cut off from the computer center application data and programs themselves, and only system personnel may have unauthorized access. Unauthorized access attempts by users are unambiguously halted.

The development of the modules within the "Protection of Information Against Unauthorized Access" subsystem described above will make it possible to solve a number of basic user access problems, as well as to use the query-order structures developed for planning, accounting, and management of the information and computing resources in the Republic Network of Computer Centers of the Azerbaijan SSR, which is being developed.

Bibliography

1. L.G. Hoffman, "Advanced Methods of Information Protection," Moscow, Mir, 1980, 264 pages.

2. D. Siao, D. Kerr, and S. Mednick, "The Protection of Computers," Moscow, Mir, 1982, 263 pages.

3. V.A. Gerasimenko and M.K. Razmakhin, "The Protection of Information in Computer, Information, and Control Systems and Networks," ZARUBEZHNAYA RADIOELEKTRONIKA, No 3, 1985, pp 41-60.

4. N.A. Krinitskiy, G.A. Mironov, and G.D. Frolov, "Programmirovaniye i algoritmicheskiye yazyki" [Programming and Algorithmic Languages], Moscow, Nauka, 1975, 496 pages.

Computer Factory: Reality and Prospects

907G0087 Kishinev SOVETSKAYA MOLDAVIYA in Russian No 12 Jan 90 [Signed to press 16 Jan 90] p 3

[Article by Moldavian Telegraph Agency]

[Text] A discussion of the real problems associated with the construction of the Kishinev personal computer factory was the subject of a meeting of the directors of party and soviet organs, and a number of interested union and republic ministries and departments. The meeting was held in the executive committee of the Kishinev City Soviet of People's Deputies. The meeting was led by the secretary of the central committee of the communist party of Moldavia, I. T. Gutsu.

The chairman of the Kishinev City Executive Committee, V. G. Dobrya, presented some information. He noted that many problems arose in the construction of this factory, and it was not so easy to solve these problems. At first, construction went on at a good pace, and there was optimism; but now, there is a danger that the factory may become one of the most expensive factories, not just in the republic, but in the country. From the estimate of 1300 million rubles provided, all but 130 million has been used. A great deal of excavation work has been done, the foundations laid, and the framework erected. A significant reduction in centralized government capital investments for 1990 threaten the plant. As a result, new problems arose with the raising of the objects of social and cultural life, the utilization of the profits of the future enterprise, and compensation for the deficit of water for the agricultural and drinking needs of Kishinev and a number of other problems, which await solutions by the factory administration and the city authorities. To this is added the ambiguous public opinion of the residents of the republic, which also need to be clarified.

All of these issues were examined at the city soviet session. The deputies proposed several alternatives for the continuation of construction, and with the help of scientists, and considering the conclusions of a competent independent commission organized by the city executive committee, it remains to make the correct choice.

The participants at the meeting carefully analyzed these alternatives, and weighed all the pros and cons. Many questions were dismissed by the chairman of the commission, member-correspondent of the Moldavian SSR Academy of Sciences N. R. Andronatiy, who is studying the prospects for the construction of the computer factory. On the behalf of a large group of authoritative scientists, he announced with complete responsibility that the only right solution was to continue construction.

"This conclusion of the commission," he said, "proceeds not only from the fact that personal computers are sorely needed in the all-union market and will bring real profits to the republic, but also from the fact that the majority of problems threatening the advisability of construction have already been solved, or the correct ways of solving them have been found. Cooperations with advanced foreign firms can be set up so we can more quickly reach full power, 150 thousand computers per year."

As for setting up ecologically pure production, which has been discussed repeatedly at various levels, there are already projects for the construction of clean buildings, and good alternatives have been proposed for the use of industrial wastes and compensation of the water deficit. If any objections remain from the Ministry of Health and the State Committee on Environmental Protection, they are not substantial, and will be taken in to consideration. Overall, as the analysis of the scientists showed, the computer factory is not a threat to the environment.

The work force problem is also being resolved, which due to the lack of necessary information, has long fomented the public, threatened with a new wave of migration. The factory administration, and the appropriate ministries and departments will train specialists from a number of residents of Kishinev and republic residents at the country's centers and abroad, as well as at the enterprises of foreign firms. They will also use, as a result of conversion and the introduction of cost accounting, production organizers and workers from other related enterprises of the city.

"Only the united efforts of republic and union ministries and departments will make it possible to solve all the problems in their common interests," noted V. A. Mikhaylov, a consultant of the central committee of the communist party of the Soviet Union. "Of course the specifications of the proposed machines do not match the latest models of foreign computers, but their level was evaluated to be high enough, by a government commission, and the computer will undoubtedly be in great demand on our domestic market. And if we use wisely the power which accompanies the conversion, then it would be possible to create in Kishinev a powerful computer center, which is very desirable and promising."

In the course of the constructive exchange of opinions, meeting participants came to a common conclusion: the construction of the Kishinev personal computer factory must be accelerated, based on active cooperation with foreign firms; production must be kept ecologically clean; and production must be kept at a high technical level.

The following participated in the meeting: sector leader of the central committee of the communist party of the Soviet Union P. K. Ishutin; deputy minister of the radio industry of the USSR E. R. Filtsev; member-correspondent of the USSR Academy of Sciences, general director of the MNTK [not further expanded] "Personal computers," I. A. Mizin; director of the socioeconomic division of the central committee of the communist party of Moldavia K. A. Tampiza; appropriate employees of the USSR Gosplan; and the directors of a number of ministries and departments of the republic.

Modems in Information-Computing Systems

907G0076A Moscow NOVOYE V ZHIZNI, NAUKE I TEKHNIKE: VYCHISLITELNAYA TEKHNIKA I YEYE PRIMENENIYE in Russian No 12, December 1989 pp 29-40

[Article by A.M. Bograd, L.G. Izrayilson and V.B. Sadovskiy; first paragraph is NOVOYE V ZHIZNI, NAUKE I TEKHNIKE: VYCHISLITELNAYA TEKHNIKA I YEYE PRIMENENIYE introduction; underlined passages are rendered in Latin alphabet in the original]

[Text] Nowadays our readers (including those far removed professionally from telecommunications) might be interested in the design, application and improvement of modems and other data transmission devices. While by no means laying claim to giving complete and thorough answers, we shall try to satisfy their curiosity to the best of our ability. We shall try to cover some aspects related to the terminology, standardization, basic operating principles and prospects for modem development. Not only can familiarity with this field of technology satisfy one's natural curiosity, but it also will help call one's attention to its numerous possible applications.

The term "information science" has been so much in general use that it even appears in the name of a USSR Supreme Soviet Commission. Let us check an encyclopedic dictionary: "Information science is a science field that studies the structure and general properties of scientific information and problems related to its acquisition, storage, retrieval, processing, transformation and application in various spheres of activity."

There is already talk and writing of the coming "age of information science," probably meaning that its influence has become tangible in all spheres of human activity and that the expansion of this influence is irreversible.

It seems it is not without reason that computer technology professionals (developers, manufacturers and programmers) are reaping the well deserved laurels of pioneers who have opened this remarkable age for us all.

The authors of this article, who have spent over 20 years in modem development, also include themselves among the "discoverers" of the new age and attribute to themselves and their colleagues at a number of enterprises involvement in at least "acquisition" and "distribution" of information.

"The marriage between a computer and communication facilities has been consummated. The wedding ceremony has been conducted, the honeymoon is behind them, and the couple is beginning to feel ever stronger how dependent they have become on one another" wrote R. Fano as early as 1972 [1].

Using communications (namely, data transmission) facilities, information sources, processing devices and consumers are combined into a single whole. Apparently, many people, regardless of their professions, have already encountered automated systems that include means for data transmission (for instance, when purchasing railroad or airline tickets).

If one believes in the inevitability of the advent of the "age of information science," then one should anticipate that very shortly data transmission devices will become consumer products and that automated supply of various types of information to the population will become an advertised service.

For information-computing systems to function properly, one should have the capability to connect territorially separated information sources and information consumers (e.g., computers). Transmission of binary information (data transmission) is conducted using special devices—modems, for which there is a home-grown term: signal conversion devices (UPS). A modem is a combination of a modulator (at the transmitting end) and demodulator (at the receiving end).

A modem consists of a transmitter, which performs unambiguous conversion of an initial binary sequence into a signal that can be used for transmission via a communication channel, and a receiver, which performs the inverse conversion. The current diversity of modems is determined by two factors—transmission rate, i.e., the number of binary symbols per unit of time (bit/s) and characteristics of the transmitting medium, such as the communication channel service band, permissible level of the transmitted signal and noise power. Modem complexity and hence price mainly depend on the efficiency with which they use communication channels, i.e. on the parameter equal to the ratio of the transmission rate to a communication channel bandwidth (specific data transmission rate). Thus, at the specific transmission rate over 1 bit/(s x Hz) a modem must be equipped with a system of adaptive correction of intersymbol distortions (ACID) to compensate for the uneven character of the frequency response of the communication channel transmission factor.

The requirement for compatibility of modems manufactured by various companies has led to the need to unify modem interfaces with terminal data equipment and communication channels. To solve these problems, the International Consulting Committee on Telegraphy and Telephony (CCITT) develops recommendations standardizing basic modem characteristics (transmission rate, the method for forming the transmitted signal and permissible range of variation of the level and frequency of the incoming signal) and interface circuits with data terminal equipment (DTE).

The progress in IC manufacturing has a considerable effect on the expansion of functional capabilities of modems, i.e., on forming modem "intelligence" and reducing cost and size parameters.

Stages in Development of Equipment for Transmission of Binary Information

According to classification in [2], modem history can be reduced to four periods.

The first period began in 1919. It was characterized by developing and placing in operation low-speed equipment for transmission of binary information (telegraphy) via long communication channels.

The second period began in the 1950s. Increased demand, knowledge and speed formed the motto of that period. Data transmission theory was developing rapidly, and new methods for modulation of and compensation for interference effects were proposed and implemented. A specific transmission rate of up to 1.5 bit/(s x Hz) became the norm, and modem designs with a rate of up to 3 bit/(s x Hz) were proposed. Transmitting data via a VF (telephone) channel with a 300-3,400 Hz service band was equivalent to a transmission rate of 9,600 bit/s.

The third period covered the 1970s. The CCITT developed a series of recommendations (series V), which standardized basic characteristics of modems operating via VF channels at a specific rate of up to and including 3 bit/(s x Hz) and modems operating via primary group paths (a 60-108 kHz bandwidth) at a rate of 48 kB/s. Effective methods for compensation of interference effects (some of them will be discussed later) were proposed and, above all, implemented. Data transmission via VF channels with two-wire termination at a rate of up to 3 bit/(s x Hz) was achieved. Developments aimed at a further increase of the transmission rate were conducted.

The fourth period began in the late 1970s. During this period, modems with the specific transmission rate of up to 6 bit/(s x Hz) (transmission rate of up to 19.2 kB/s via a VF channel) have been practically mastered. Algebraic signal processing methods have been implemented in data transmission theory.

Data transmission theory and practice demonstrate that increasing the specific transmission rate has been the main incentive for modem development. C. Shannon demonstrated that in the presence of fluctuation noise in a channel the specific transmission rate is limited to $\log_2(1+P_s/P_n)$, where P_s and P_n are the transmitted signal power and noise power, respectively. For instance, for a VF channel it is approximately equal to 8-10 bit/(s x Hz).

However, when operating via real-life communication channels, one should take into account not only fluctuation noise, but also other interference factors.

Characteristics of Transmitting Medium

The data source and data recipient are connected to a data transmission channel, the block diagram of which is shown in Figure 1.

An error protection device (EPD), which is part of the data transmission equipment (DTE), is an optional DTE element, which can be combined with the modem, if necessary.

A communication channel in an analog transmission system (a transmission system with frequency multiplexing) occupies a certain bandwidth. The overwhelming majority of modems are designed for operating via VF channels. It is therefore natural to become acquainted with the interference factors, namely, distortion and noise in a VF channel, that limit the transmission rate and reception quality of transmitted signals. These include nonuniformity of the frequency response of the communication channel transmission factor, i.e., deviation of the amplitude-frequency characteristic from its value measured at 1,020 Hz; deviation of the value of group propagation time (GPT) from its value measured

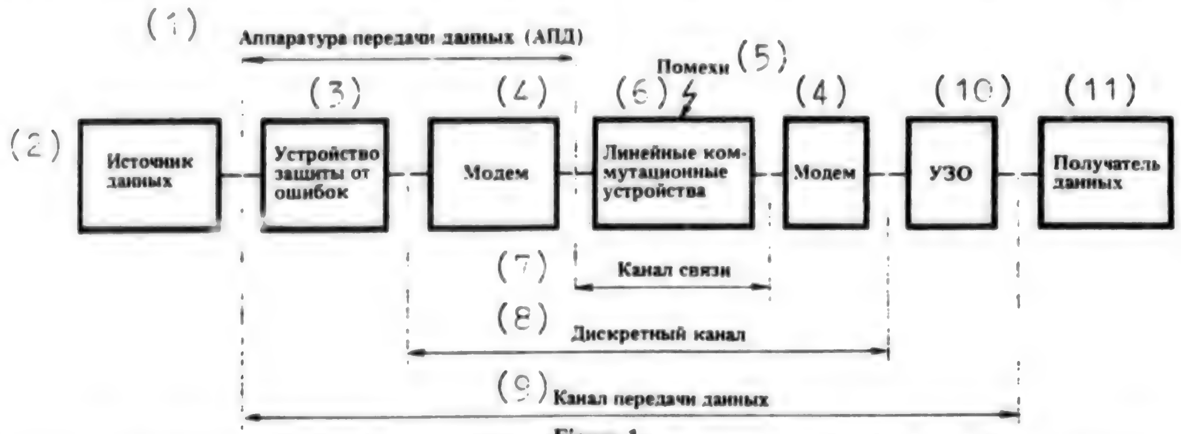

(1) Аппаратура передачи данных (АПД)
(5) Помехи
(2) Источник данных
(3) Устройство защиты от ошибок
(4) Модем
(6) Линейные коммутационные устройства
(4) Модем
(10) УЗО
(11) Получатель данных
(7) Канал связи
(8) Дискретный канал
(9) Канал передачи данных

Figure 1.

Key:—1. Data transmission equipment (DTE)—2. Data source—3. Error protection device (EPD)—4. Modem—5. Noise—6. Line switching devices—7. Communication channel—8. Discrete channel—9. Data transmission channel—10. EPD—11. Data recipient at 1.9 kHz; a change in signal frequency in the channel; fluctuation noise; pulse noise; momentary level drops, etc.

Nonuniformity of the GPT characteristic (GPTC) considerably affects transmission quality, because it causes intersymbol interference (ISI) in the transmitted sequence of data signals, which can lead to errors even in the absence of interference. High GPT nonuniformity is due to the fact that the main purpose of VF channels is transmission of speech signals, and the quality of their reception is not affected by GPT nonuniformity.

Figure 2 shows averaged GPT values of VF channels in the case of one, five and ten transits (N). It also shows standards per CCITT Recommendation M.1020.

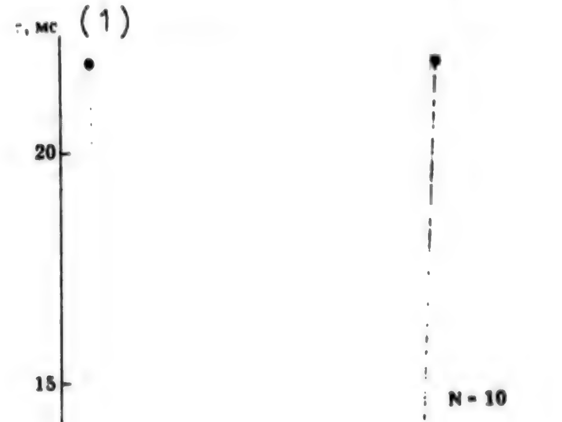


Figure 2.

Key:—1. ms—2. kHz

Momentary interruptions and pulse noise are main sources of data transmission errors [3].

Frequency change in a channel. Distortions of this type are manifested in the form of a small change (shift) of the absolute value of frequency of each component of the transmitted signal spectrum at the channel output relative to frequencies of these components at the channel input. The presence of this type of distortion is due to a mismatch of frequencies of the generating equipment of the linear path. According to standards, frequency change should not exceed 5 Hz.

Practice demonstrates that fluctuation noise power changes in time over a wide range and depends on the length of a communication channel. Mean value of the level of fluctuation noise in a 12,500 km long channel should not exceed 37.5 dBm.

A complete list of the electric parameters for communications channels and methods for parameter measurement is presented in [4].

The types of distortion and noise discussed above to a large extent determine methods for forming and processing transmitted data signals used in modems.

Modem Structure

A typical generalized block diagram of a modem is shown in Figure 3. In the modem transmitter, the following operations are performed:

—matching, using interface devices, parameters of signals arriving at the transmitter from information sources and converting them to a form convenient for processing;

—the coder transforms input binary information sequence coming from DTE into a sequence of signals according to the type of modulation;

—the modulator carries the spectrum of the transmitted signal to the communication channel bandwidth (when using a bandwidth-limited channel); and

—the matching device performs necessary amplification and filtering of a modulated analog signal.

At the receiving end a signal undergoes reverse transformations.

In the simplest case, a transmitted sinusoidal signal, called carrier oscillation, can be modulated by a binary sequence by changing the amplitude (amplitude modulation), frequency (frequency modulation) or phase (phase modulation). In complex types of modulation one can use combinations of these parameters. Thus, for instance, in the case of quadrature amplitude modulation (QAM) amplitude and phase are changed simultaneously. In the general case, the transmitted binary information sequence is divided into groups of m symbols, and the coder transforms each group into a specific

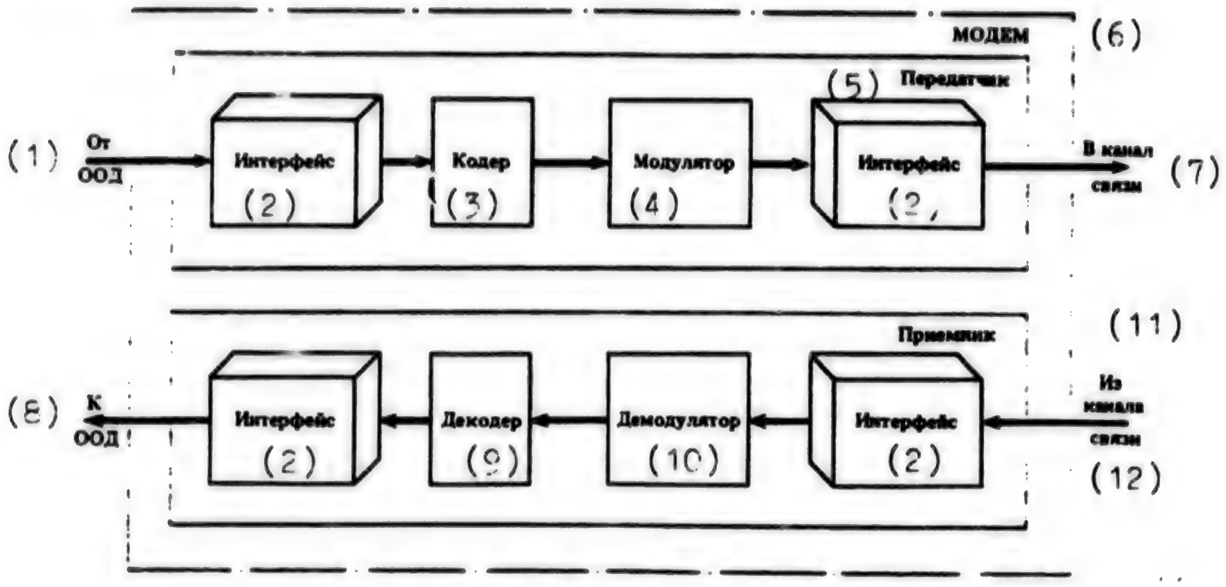


Figure 3.

Key:—1. From DTE—2. Interface—3.Coder—4. Modulator—5. Transmitter—6. Modem—7. To communication channel—8. To DTE—9. Decoder—10. Demodulator—11. Receiver—12. From communication channel

element of a data signal. Obviously, each element contains $\log_2 m$ bits of information, where m is the signal alphabet at the coder output. The choice of the modulation method depends on the required specific transmission rate, which is equal to $\log_2 2m$. Frequency modulation (FM) is used when modem simplicity and economic parameters are more important than efficient utilization of a channel frequency band. This is why FM is used in modems for data transmission at rates of up to 1200 bit/s.

The use of phase modulation (PM) makes it possible to increase the specific data transmission rate due to the expansion of the data signal alphabet at the coder output by increasing the permitted number of phase positions of the transmitted signal. Thus, for instance, in the case of double relative phase modulation (DRPM) the number of phase positions $m = 4$, i.e., $\log_2 m = 2$, and in the case of triple relative phase modulation (TRPM) $m = 8$, i.e., $\log_2 m = 3$. The term "relative" indicates that an element phase is changed relative to the phase of the preceding element of the transmitted signal. Permitted positions of the data signal at the coder output are shown in Figures 4a for DRPM and 4b for TRPM.

A further increase of the specific transmission rate is possible if combined types of modulation are used.

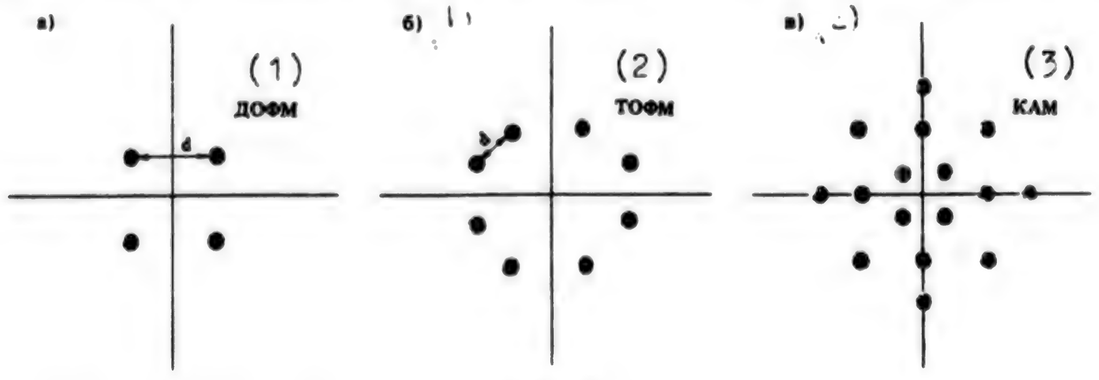


Figure 4.

Key:—1. DRPM—2. TRPM—3.QAM

Figure 4c shows permitted positions of the data signal at the coder output in the case of quadrature amplitude modulation (QAM) for m = 16, i.e., $\log_2 m = 4$. At present, using QAM, a transmission rate of 14.4 kB/s via a leased VF channel ($\log_2 m = 6$) is being mastered, and experimental studies on a transmission rate of 19.2 kB/s ($\log_2 m = 8$) are conducted. Figure 4 shows that the increase in the specific transmission rate ($\log_2 m$) is achieved by reducing distance d (in the Euclidean sense) between permitted data signal positions. The magnitude of d determines noise immunity of the received signal, i.e., the modem receiver's capability to accurately reproduce the transmitted information sequence in the presence of noise in a communication channel. If the value of noise is less than d/2, then a transmitted data signal is estimated accurately in the receiver decoder.

The ultimate theoretically achievable value of noise immunity for each type of modulation is called its potential value. Comparison of the actual noise immunity of a modem to its potential noise immunity makes it possible to assess the quality of modem implementation, because potential noise immunity only depends on d and does not depend on the reception method. Noise immunity is determined by the value of the signal/noise ratio at the receiver input at a given error factor K_{err}. Analytical expressions for computing potential noise immunity for various types of modulation are presented in [5]. These expressions were used to derive K_{err} as a function of the ratio of signa. power to fluctuation noise power (P_s/P_n. These functions are shown in Figure 5 for DRPM, TRPM and QAM. It follows from Figure 5 that an increase in m, i.e. an increase in the specific transmission rate, leads to a reduction in potential noise immunity. For instance, at $P_s/P_n = 24.5$ dB, which is typical for a VF channel up to 12,500 km long, and when using QAM, the error factor will be less than 10^{-7}. At the same time, a modem implementation error leads to lower noise immunity (compared to potential noise immunity). Usually the reduction does not exceed 2-3 dB. Therefore, when working at a rate of 9600 bit/s via VF channels, one can expect $K_{err} = 10^{-4}$-10^{-7}.

ISI actually decreases the value of d at reception, which is equivalent to a reduction in noise immunity of the received data signal. At a specific transmission rate over 1 bit/(s x Hz), ISI is the main factor that determines data transmission fidelity in a VF channel.

When frequency characteristics of a communication channel are known, a signal spectrum can be formed in such way that the frequency characteristic of the entire transmission path (transmitter, communication channel and receiver) satisfies certain conditions under which ISI is absent.

These conditions were first formulated by Nyquist [6]. However, in practice characteristics of the communication channel are unknown and moreover, they change over time; therefore, to decrease ISI at a modem decoder input, formation of the data signal spectrum and correction of the received signal are used. The data signal

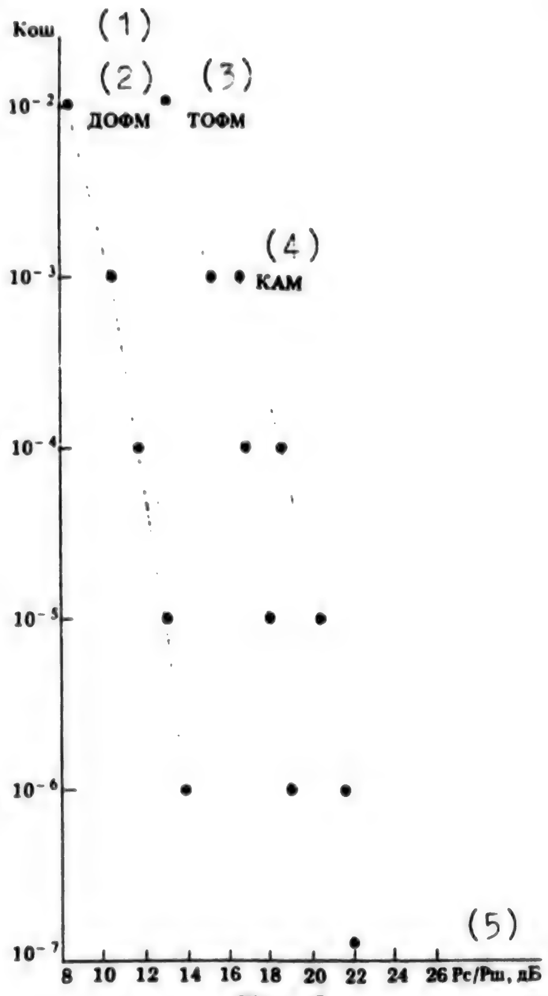


Figure 5.

Key:—1.—2. DRPM—3. TRPM—4. QAM—5. dB

spectrum can be formed in the transmitter and/or receiver by using digital or analog filters. It has been demonstrated theoretically that if fluctuation noise is the only interfering factor in a communication channel, the task of forming a signal spectrum must be shared equally by the transmitting and the receiving filter. However, to simplify modem structure, the data signal spectrum former is often implemented in a transmitter, using a nonrecursive (without feedback) digital filter.

For ISI correction, correctors with manual and/or automatic tuning may be used. A manual corrector tuning method does not provide high accuracy of ISI correction and is therefore used mostly in modems with a low specific transmission rate, under 1 bit/(s x Hz). Correctors using an automatic tuning method make it possible to adapt the corrector transmission factor when communication channel characteristics change. ACID have been widely used in modems, because they make it possible to provide a high degree of ISI compensation. A typical ACID structure is a nonrecursive digital filter. Such arrangement of an ISI correction system matches well a microprocessor-based modem implementation.

There are synchronous and asynchronous transmission methods. In the case of asynchronous transmission, the duration of data signal elements is determined by moments of changes of symbol values in the transmitted information sequence, while in the case of synchronous transmission element duration is constant. Asynchronous transmission methods are mainly used in modems with frequency modulation. Synchronous transmission methods provide higher noise immunity, because due to synchronization of elements of a received signal, a signal at the decoder input is estimated at moments corresponding to the least distorted portion of the signal. Synchronization of elements of received signals is formed based on nonlinear processing of a signal arriving from a communication channel, and the phase and frequency of synchronization signals are usually fine-tuned by indirect action on the master oscillator, using pulse addition or subtraction from the generated pulse sequence.

To ensure high noise immunity in the case of phase modulation (or QAM), the modem receiver uses coherent detection of the received signal. Therefore, it is necessary to form at the reception end a coherent oscillation equivalent to the master oscillation at the transmitting end, taking into account the frequency shift and phase jitter of the signal in the communication channel. In modems with a high specific transmission rate, coherent oscillation is separated by analyzing the magnitude of a signal phase shift at the decoder output. A more detailed description of modems is presented in [5].

Types of Modems

Modems are classified depending on their transmission rate and operating mode. When transmitting data via VF channels, the following operating modes are possible:

—via switchable or nonswitchable channels;

—via a communication channel with a two- or four-wire termination; and

—a duplex, semiduplex or simplex transmission mode.

Switchable channels (TRPM general-use telephone channels) are usual channels that provide automatic establishment of connections of telephone subscribers using switching equipment of relay or electronic ATX. In this case, the length and parameters of the communication channel that connects two modems vary over a wide range, depending on the availability of free lines in ATX group equipment when a subscriber's number is dialed. Therefore, modems operating via nonswitchable (leased) channels (at the same transmission rate) are considerably simpler and provide higher transmission fidelity.

In the case of four-wire termination of a communication channel, transmission and reception are separated mechanically, because they are conducted over different wires, while in the case of two-wire termination the function of separating transmission and reception signals is given to the modem. Usually this function is performed by a built-in differential system similar to that used in a telephone set.

A modem usually operates in one of the following three modes: duplex, semiduplex or simplex. In duplex mode, modems connected to each other can transmit information to each other simultaneously. In simplex mode, data transmission is only conducted in one direction, and in semiduplex mode turns are taken in transmission. In practice, the most widely used mode is a duplex transmission mode via a four-wire nonswitchable and two-wire switchable VF channel.

A modem can be connected to a VF channel either directly or by means of acoustic connection. In the case of acoustic connection, a modem includes an acoustic coupler (rubber cups placed on the telephone handset). An acoustic coupler amplifies the transmitted signal to an audible level detected by the telephone handset microphone. Due to the effect of external noise, this connecting method is used in modems with a transmission rate of up to 1,200 bit/s.

Table 1 shows basic parameters of modems designed for operation via VF channels, based on CCITT recommendations.

Modems are made either as built-in into a computer or stand-alone (external) devices. Built-in modems are inserted into expansion slots, while external modems are installed next to a computer and connected to it via an interface (for instance, an S2 interface).

A built-in modem is slightly less expensive than an external one, and no interface is needed for its operation. However, an external modem is more convenient in operation, because it is equipped with devices for displaying control signals, the status of the transmission channel etc. Moreover, external modems are more flexible because their parameters are defined by users rather than by the computer developer.

Table 1

CCITT Recommendations/ Parameters	V21	V22	$V22_{bis}$	V23	V26	$V26_{bis}$	$V26_{tr}$	V27	V_{bis}	$V27_{tr}$	V29	V32	V33
Maximum transmission rate, bit/s	300	1200	2400	1200	2400	2400	2400	4800	4800	4800	9600	9600	14,400
Channel termination type	2	2	2	2	4	2	2	4	4	2	4	2	4
Type of modulation	FM	DRPM	QAM	FM	DRPM	DRPM	DRPM	TRPM	TRPM	TRPM	QAM	QAM	QAM
Channel correction	C	A	A	C	C	C	A	M	A	A	A	A	A
Operating mode	AS	AS, S	AS, S	AS	S	S	S	S	S	S	S	AS, S	S
Communication network	GUTC	GUTS	GUTS	GUTC	LTC	GUTC	GUTC	LTC	LTC	GUTC	LTC	GUTC	LTC

Legend:

GUTC—general-use telephone channel;

LTC—leased telephone channel;

AS—asynchronous;

S—synchronous

C—constant;

M—manual;

A—automatic.

It follows from the above that when choosing a modem one should pay attention to the following technical and economic parameters:

—the transmission rate and type of modulation. They can be selected using Table 1;

—noise immunity and transmission range;

—the presence and type of interfaces;

—overall dimensions and power consumption;

—service simplicity;

—price.

The availability of recently published reference books [7, 8] makes a detailed description of interface circuits unnecessary. We shall therefore limit ourselves to a general description of modem connection to an information network.

The process of establishing a connection between various subscribers of an information-computing network begins with dialing (calling) a subscriber. A call can be made either by the subscriber him- or herself (manually), using a dial, or automatically by DTE with the help of a computer or an automatic calling device (ACD), which according to CCITT Recommendations V.24 and $V.25_{bis}$ is included in DTE. In the manual method for establishing a connection when transmitting data via a GUTC, the operator at point A manually calls a telephone number at point B. The operator at point B answers the call by taking the handset off the hook, establishing a connection between points A and B. After verbal verification that a correct connection has been established, both operators (at points A and B) connect their modems to a GUTC line.

Automatic establishment of a connection for data transmission can be performed by a computer that calls the needed number automatically. This is called an automatic calling operation. Now there are modems with automatic calling capability. A computer connected to such modem uses an interface according to CCITT Recommendation V.24/V.28 or RS-232-C for automatic calling and data transmission operations. CCITT Recommendation $V.25_{bis}$ standardizes the automatic calling procedure for asynchronous-synchronous modems with automatic calling capability. Usually, equipment at point B sends an automatic answer to an automatic call that has come from point A.

A computer or modem can automatically answer a call placed manually or automatically. This is called an automatic answering operation. A modem and interface manufactured according to CCITT Recommendation V.24/V.28 or RS-323 [per original]-C are used as hardware.

Methods for Improving Transmission Fidelity

Values of error factors by elements depend to a large degree on the status of the communication channel. Therefore, in order to improve transmission fidelity, one should implement organizational and technical measures aimed at improving communication channel parameters (by replacing obsolete equipment, periodically monitoring basic parameters of the communication channel, etc.). Within the framework of this article, it is more natural to discuss methods for improving transmission fidelity based on introducing redundancy into the transmitted data signal. There are two basic methods for introducing redundancy. In the first method, additional binary symbols are added to the transmitted binary information sequence. In the second method, redundancy is introduced in a modem coder by expanding the signal alphabet (increasing m). In practice, for modems with a data transmission rate over 2,400 bit/s either one of these methods or a combination of both may be used. Using the first method, in order to improve fidelity, a sequence of information symbols is coded prior to arriving from DTE to the modem transmitter input. The information flow is split into n-element blocks, and r additional binary symbols are added to each block in accordance with a certain algorithm. The choice of the type of the algorithm (code) and amount of redundancy (ratio r/n) depends data transmission fidelity requirements for specific connections.

By introducing redundancy for direct transmission of information, not all possible $2^{(n + r)}$ code combinations are used, but only some of them, called permitted combinations, while the rest are prohibited ones.

If during transmission via a communication channel a permitted combination becomes a prohibited one due to noise, then at the receiving end it becomes possible to detect the presence of errors. If the number of prohibited combinations is increased, then errors may not only be detected, but corrected as well.

Coding algorithms can be either hardware- (as shown in Figure 1) or software- (using a computer) based. However, in the latter case it is necessary to use a high-speed computer.

The essence of the second fidelity improving method is to construct signal spaces that ensure the best signal discrimination at a given useful information transmission rate.

In a modem transmitter coder, a signal is formed whose alphabet has many more permitted positions than those shown in Figure 4. The number of permitted signal positions can be 128 and more. Using convolution (lattice) coding in the algorithm coder makes it possible to increase distance d and hence substantially improve transmission fidelity.

Recently, this method has been widely used abroad in synchronous modems operating at speeds from 9,600 to 19,200 bit/s.

Modems using this fidelity improvement method have been standardized in CCITT Recommendations $\underline{V}$.32 and $\underline{V}$.33. The modems have tangible advantages:

—reliable data transmission via a switchable telephone network at a rate of up to 9,600 bit/s can be achieved. However, one should keep in mind that there have been no reports yet on high-speed data transmission via domestic switchable VF channels;

—personal computer users can transmit information at a much higher speed, using currently available communications software.

The number of published works analyzing various aspects of application of convolution coding in modems meeting CCITT Recommendations $\underline{V}$.32 and $\underline{V}$.33 has been increasing exponentially. However, most of them are based on [10], where this method was first proposed.

Functional Capabilities of Modems

Modem characteristics depend to a large degree on user requirements:

—efficient utilization of communication channels; and

—the level of service.

At present, the problem of improving the efficiency communication channel use is solved by increasing the specific transmission rate and/or compressing transmitted binary information.

Transmission rate is increased by expanding a transmitted signal alphabet (increasing m), combined with the use of high-efficiency methods for ISI compensation and separation of synchronous oscillations.

Modem structures with compression devices described in [11] make it possible to improve the efficiency of communication channel use approximately twofold.

The very first modems performing only basic modem functions and essentially did not recognize directives coming from DTE. This considerably limited their implementation in information-computing systems, thus reducing the rate of modem production increase.

Modern modems are much "smarter." They execute a fairly broad set of instructions they exchange with the computer. The standard developed by Hayes Microcomputer Products (USA) is an example of such set. The Hayes set of instructions, also called the AT set of instructions (the letters AT initiate modem operation) makes it possible to considerably improve the level of service.

At the start of a communication session, the receiving modem adapts itself to the data transmission rate of the transmitting modem and then determines if the latter meets a CCITT Recommendation series $\underline{V}$. If it does not, the modem determines whether the transmitting modem operates in a synchronous or asynchronous mode and accordingly tunes itself automatically to the required mode. Communications between series $\underline{V}$ modems are conducted in synchronous mode, which makes it possible to create a transparent communication channel between personal computers and mainframes.

Basic Hayes instructions are listed in Table 5 in V.V. Malezhenkov's article.

Using this instruction set, one can control and change modem operating mode, for instance, specify the optimum transmission rate, set the number of call-waiting seconds etc. Hayes instructions are described in greater detail in [11].

Status of the Modem Market

The world modem market continues to grow rapidly. Thus, for instance, in the USA sales of modems operating via VF channels increased 20 percent in 1986. It is predicted that by 1990 up to 18 million modems will be sold in the USA and that over 60 percent of personal computers will have built-in modems.

There is a stable trend of increasing the share of miniature modems that use one VLSI chip and can be included in various devices (e.g., a computer). Their prices vary from $40 (simple low-speed modems) to $12,000 (modems that operate at speeds up to 19,200 bit/s and provide high level of user service).

When analyzing the status of the modem market, one can identify the following development trends based on potential user needs:

a) obvious progress in modem miniaturization;

b) expansion of functional capabilities of modems. The fact that an ever larger number of new models have protection from unauthorized access to databases is but one example. In this case, data transmission only takes place after verifying passwords. In addition, all modem access attempts are recorded;

c) an increased number of digital modems operating that use microprocessors.

There are reports of digital modems at a rate of 2.048 Mb/s in the secondary and 6.3 Mb/s in the tertiary group channel.

In addition, a number of foreign companies are mastering production of modems for data transmission via fiber-optic communication lines. Such modems are already on the market; however, their share in the total modem production volume has been low thus far.

Bibliography

1. Fano, R., "O roli organizatsii sluzhby peredachi dannykh v zhizni obshchestva. Sistemy peredachi dannykh i seti EVM" [Role of Organization of Data Transmission Service in Society. Data Transmission Systems and Computer Networks], Moscow: Mir, 1974.

2. Pahlavan, K. and Holsinger, J.L., "Voice-Band Data Communication Modems - A Historical Review: 1919-1988," IEEE COMMUNICATIONS MAGAZINE, Vol 26, No 1, 1988.

3. "Kanaly peredachi dannykh" [Data Transmission Channels], Moscow: Svyaz, 1970.

4. GOST [State All-Union Standard] 21655-87. "Kanaly i trakty magistralnoy pervichnoy seti yedinoy avtomatizirovannoy sistemy svyazi. Elektricheskiye parametry i metody izmereniy" [Channels and Paths of Primary Trunk Network of Unified Automated Communication System. Electrical Parameters and Methods of Measurement].

5. Danilov, B.S. et al., "Ustroystva preobrazovaniya signalov peredachi dannykh" [Devices for Conversion of Data Transmission Signals], Moscow: Svyaz, 1979.

6. Nyquist, H., "Certain Topics in Telegraph Transmission Theory," AIEE TRANS., Vol 47, April, 1928.

7. Myachev, A.A., Stepanov, V.N. and Shcherbo, V.K., "Interfeysy sistem obrabotki dannykh" [Data Processing System Interfaces], Moscow: Radio i svyaz, 1989.

8. Jennings, F., "Prakticheskaya peredacha dannykh. Modemy, seti i protokoly" [Practical Data Transmission: Modems, Networks and Protocols], Moscow: Mir, 1989.

9. GOST 20768-75. "Apparatura peredachi dannykh. Ustroystvo avtomaticheskogo vyzova UAV-TIF" [Data Transmission Equipment. Automatic Calling Device UAV-TIF].

10. Ungerboeck, G., "Channel Coding With Multilevel/Phase Signals," IEEE TRANSACTIONS ON INFORMATION THEORY, Vol IT-28, 1982, pp 55-67.

11. "New Series of Intellect Modems", PC WORLD, 1988, Vol 6, No 3.

Analog to Digital Converter

907G0086 Moscow NTR TRIBUNA in Russian No 1-2 Jan 90 [Signed to press 19 Jan 90] p 13

[Text]**Analog to Digital Converter**

A family of eight-channel, ten-bit analog to digital converters with interfaces for the series DVK personal computer and the IBM PC XT(AT):

SET-1 is a stand-alone unit with an autonomous power supply and personal computer interface. Price 2820 rubles.

SET-2 is a one-board version which goes in the DVK series personal computer. Price 1135 rubles.

SETU-10 is a one-board universal programmable interface which goes in the IBM PC/XT(AT), and includes modules for analog input and output, digital input-output, a timer, and an interrupt module. Price 2380 rubles.

SHCHIT-1 [Shield-1] is a stand-alone unit with an autonomous power supply, galvanic bypass to meet the requirements of medical electrical safety, and an interface for a personal computer. Price 3920 rubles.

SHCHIT-2 [Shield-2] consists of two stand-alone units with a 300 meter fiber optic transmission line which provides a galvanic bypass to meet the requirements of medical electrical safety. Price 6995 rubles.

The products of the cooperative can be obtained in the Instruments and Computer Equipment store at 30/43 Nakhimovskiy Prospekt, Moscow. Take the Metro to the Profsoyuznaya station.

Telephone numbers for orders: for technical questions, 415-18-97; for purchasing questions, 129-09-36.

Development of Software for Designing Expert Decision Support Systems

907G0047A Kiev MEKHANIZATSIYA I AVTOMATIZATSIYA UPRAVLENIYA: NAUCHO-PROIZVODSTVENNYY SBORNIK in Russian No 4, Oct-Dec 89, pp 34-38 [MS Received 09 Mar 89]

[Article by N.D. Vashchenko, candidate of technical sciences, V.P. Gladun, doctor of technical sciences, and Yu.L. Dryuchin, candidate of technical sciences]

[Text] The scheme for designing applied expert systems (ES) on the basis of shell systems, effective at first glance, often turns out to be hard to implement or of little use in solving complex practical problems, above all, due to insufficient development of modern means for creating and managing information bases in expert systems. In particular, as a rule, only one of the basic existing methods for representing knowledge is used in shell systems, although when solving problems a certain combination of them is usually required in practice. In this regard, along with known problems of formal representation of expert knowledge in an ES, the topical problems of integrating methods for representing and processing knowledge in an ES with traditional data bases and applied program packages are coming increasingly to the forefront.

Considering the complexity of adapting "empty" expert systems to subject fields, it is expedient to focus efforts on developing and introducing sets of standardized software and hardware for automating the processes of building applied expert systems with developed, hybrid knowledge bases, large external (in relation to the ES systems information base) data bases, and applied program libraries.

At the "Gorsystemoteknika" NPO (Kiev), the KODEKS software tools complex was developed, intended for designing applied expert decision support systems based on a personal computer. The EPROS system (V.P. Gladun, "Planirovaniye Resheniy" [Decision Planning], Kiev, 1987) was the prototype for a number of design solutions in KODEKS.

The KODEKS complex includes the following basic blocks:

1. A system for managing data bases in expert systems, which is a functionally complete software product and consists of: a data base (DB) management system, which ensures the creation and management of developed production-frame type knowledge bases; a system for managing on-line data bases (ODB), which ensures the formation and management of relational type data bases, containing the formulations of tasks (descriptions of initial situations and target conditions), as well as descriptions of intermediate situations that arise when seeking decisions in an ES; standardized software modules for the design, applicability testing, and execution of production rule variants.

2. A decision planning processor which implements a software mechanism for logical deduction and heuristic searching of multi-step solutions to problems, formulated on the basis of the data and knowledge bases. The decision planning processor is supplied with a library of strategies for searching decision paths.

3. A subsystem for explaining the processes and results of a decision search in the ES, which operates on two levels: for the end user, and for the knowledge engineer.

4. Systems for forming and managing catalogues containing the necessary information about libraries within the ES for applied programs and estimation functions, used to select production rules from the DB. These systems make it possible to standardize the process of forming applied program and estimation functions without changing the software system.

5. Software for connecting the ES information base management system to external data bases, particularly by using formalized methods for making specialized programs that provide for the possibility automatically to use the contents of external data bases, created in computing environments compatible with that of the KODEKS complex. Connection to external data bases is achieved via selection of relevant data from them and the writing of said data to the ODB.

Allocation in the ODB of a relatively large share of the data accessible to the user in a subject field makes it possible to restrict the field of the search for problem-solving paths, as well as significantly to raise the effectiveness of the decision search, due to utilization of data structures in the ODB, oriented toward mechanisms implemented in the ES, in order to process production-frame type knowledge.

6. A knowledge acquisition subsystem, the initial information content of which is comprised of general and task-oriented procedural recommendations for formalizing knowledge on the subject fields for the ES.

The basic distinguishing feature of the KODEKS complex is the presence of more highly developed (compared to known analogues) methods for modeling the problem environments and decision-making mechanisms, and ways to design knowledge bases which enable one to implement certain technologies for forming knowledge about classes of tasks.

On the example of the KODEKS complex, specific features of the basic components from a set of software tools—information base management systems and a decision-making processor with a library of strategies—are considered below.

Analysis of the present state of works in the field of organizational management indicates a steady trend toward a combination of the production and frame approaches to knowledge representation within the framework of specific expert systems. Usually, depending on the class of problems being solved, one of these approaches is dominant. For purposes of raising the degree of automation for the processes of forming decision algorithms (not just information retrieval functions), the production approach is often preferable.

By combining the production and frame approaches to knowledge representation in obvious form, besides supporting external graphics capabilities and facilities, the problems of knowledge processing efficiency are solved more successfully. The structural multi-level hierarchy of models (of objects and actions), supported by frames, may be useful even in cases where there is no need for it from the viewpoint of interesting features of the given class of problems. In this regard, as a rule, it is possible to introduce a hierarchy into the knowledge base according to formal attributes, for example, singling out and naming frequently repeated fragments of production rules in the frame descriptions, which permit an arbitrary nesting [vlozhennost] depth. As a result, the conciseness of description in the knowledge base is raised and search in it is accelerated.

Utilization of a strictly production approach presumes a noticeable mixing of emphases toward procedural knowledge representation. Contemporary ES (AGE, PIES, and others) offer the possibility, in the production form, of describing the change of other elements in the knowledge base and regulating the order of use of a definite type of production. Similar means of knowledge representation are reminiscent of somewhat exotic, but traditional, essentially algorithmic languages. Evidently, we should actively continue the search for justifiable compromises which, given sufficiently powerful ways to describe procedural knowledge, would not lead to a substantial loss of the basic merits of production knowledge bases: their modularity and flexibility with regard to changes, and "transparency," if even at the knowledge engineer's level.

The methods being developed for knowledge representation in KODEKS offer the following possibilities:

Description of initial data, target conditions and production rules in terms appropriate to the subject field on the basis of stereotypical (routine) expressions given by the user; stereotypes (definitions of data types), in particular, may take the form of natural language suggestions, in which indexed variables are used instead of individual words;

Standardization of production rules into operator configurations (operators) with a hierarchical structure for adequate representation of the side effects from the actions being modeled and of cause-effect relations; the operators determine the partial sequence of rule execution;

The use of an arbitrary number of positive and negative statements in the left and right parts of the production rules;

The connection of statements under conditions of applicability of operators by logical linkages of the "AND" or "OR" type;

Utilization in statements, when describing generalized production rules, of both indexed variables, as well as of the names of pre-defined classes of objects, instead of the names of specific objects;

Using production rules, addressing applied programs and standard built- in procedures, recorded in a special catalogue, i.e., the openness of the ES.

The on-line data base of the KODEKS complex has the basic features of a relational data model. The user perceives the data stored in the ODB as two-dimensional tables, whose headings are routine expressions (stereotypes) declared before the start of operation. With the help of knowledge base management systems and the ODB, the user determines the operations that should act on the ODB tables. On the other hand, the ODB dictionary is connected through obvious relations to all words given in standardized tables. Thus, the features of an associative approach, which ensure the efficient allocation of ODB fragments (given relations with the required combination of words) for rapid searching and processing of relevant data, are implemented at the internal level.

It should be noted that, since the basic purpose of the ODB is to store an instant snapshot of the real object or process being modeled, as well as the results of modeling actions using a planner, then, compared to traditional technology for designing and managing data bases, the role of the user himself in managing the ODB acquires secondary significance. In fact, the processes of selecting and renewing data in the ODB are simplified for the end user.

In solving the transformation tasks, the ODB is managed by the planner. It is precisely the planner, in interpreting the production rules during applicability testing and execution of them, who essentially uses operations with relations that are typical of the relational model. These operations are usually given by the production rules vaguely, for example, as a description, under applicability conditions, of a rule for two relations, related to each other by variables that may correspond to assigning the operations of projection, intersection, or selection.

The presence of developed structural ties and generality of the basic working components of the ODB and knowledge base make it possible, in performing similar operations, to reduce random data surplus to a minimum.

In the systems of the KODEKS complex, currently no specific approach is reflected for existing approaches to representing and processing type ambiguities in ES knowledge bases, since the first (base) version of KODEKS should, by design, unite the most common methods for designing ES. Thanks to the well-developed methods for knowledge representation and processing, which the KODEKS complex offers, the general "body" of software, on the basis of which applied expert systems will be built, can be relatively easily built upon using specific procedures for revealing and processing ambiguities of different types. These special procedures can be developed relatively independently, can be included in the ES library of built-in procedures according to general scheme, and can be used to expand the sphere for efficient application of the standardized systems of the KODEKS complex.

At the present time, the role of formal logic in knowledge representations and in designing decision search mechanisms is being considerably re-evaluated. Increasingly more often, ES developers are trying to use hybrid schemes which incorporate the methods of formal deductive conclusion, and methods for implementing probable reasoning and inductive and abductive conclusions.

Standard deductive conclusion schemes are monotonous in the sense that the axioms (initial facts) and derived statements in them are not reconsidered (the conclusions being produced are additive), and the qualities of axioms and deduction rules do not change in solving problems. Meanwhile, natural expert reasoning in solving the absolute majority of problems, as a rule, are fundamentally non-monotonous in nature: various assumptions are constantly introduced and ruled out with the extensive utilization of probable arguments, heuristic assessments, and rules. Evidently, if highly inventive steps are not taken, then, due to the monotonous nature of the conclusion, the class of problems to be solved will be limited and the design of expert systems will be complicated on the whole.

Taking the given considerations into account, a decision-making processor with a library of decision path search strategies is included in the structure of the KODEKS complex. In this regard, decision support in the general case presumes a combination of mechanisms for non-monotonous logical deduction and for heuristic search of multi- step solutions to problems, according to their formulations based on the knowledge base.

Strategies are definite schemes for a decision search, implemented in KODEKS by software and aimed at formal classes of problems. In particular, KODEKS includes the basic strategies of direct and backward search (i.e., from the initial situation to the goal and vice versa), not depending on interesting features of the problem areas.

The decision-making processor itself does not establish any kind of of decision search scheme, but does include the basic standardized procedures and interactive methods needed when using the strategies (procedures for forming and tracking the search column, for selection of the best variants on the basis of evaluation functions, the selection of evaluation functions, strategies, their parameters, etc.). Such modular division of tools makes it relatively easy to expand and modify the library of strategies.

The standardized representation of facts in an ODB, independent of specific types of strategies, makes it possible to combine existing strategies for direct and backward searching with fixed evaluation functions, organizing a bi-directional search on different variants.

Several technical characteristics of the KODEKS complex are given below:

Base programming language—C;

Menu-oriented interface systems and a help mode are used in all complex subsystems;

The maximum dimensions of the ODB and knowledge base are not limited by the KODEKS complex software;

In the event of insufficient main memory in the complex subsystems, a dynamic exchange with external memory is stipulated;

The KODEKS complex is implemented according to the principle of modularity, which makes it possible to use different configurations of it to solve specific classes of problems and to further develop software.

The hardware and software used:

A personal computer, compatible with the IBM XT PC;

MS-DOS 3.1 or higher version operating system;

A minimum main memory size (for loading the complex) of 512 K and two floppy disks.

The objects for application of the KODEKS complex include scientific research and design organizations working to automate the processes of solving poorly formalized tasks in the fields of organizational management, scientific research, and design. The users of the KODEKS complex are those who develop applied expert systems: applied programmers, experts, and knowledge engineers.

UDC 621.73:658.512.2.011.56

Principles for Development of Expert Systems for Hot Closed-Die Forging

907G0111A Moscow KUZNECHNO-SHTAMPOVOCHNOYE PROIZVODSTVO in Russian No 1, Jan 90 pp 5-8

[Article by T. N. Penchev, candidate of technical sciences; manuscript received 9 Dec 89]

[Excerpts] The necessity of developing expert systems with computer-aided design for hot closed-die forging (GOSh) has been developed by the use of a large amount of empirical knowledge, which, more often than not, cannot be represented in a formalized fashion. Expert systems in this field are being developed for the first time and the basis of the possible principles for their development has not been fully researched. Thus, for example, in the USA, one system [1] is being based on the software product ALPID to use the finite element method [MKE] with large plastic deformations. The finite element method is used both for verifying the correctness of the develpment of the forging transitions and for additional research, for example, for determining the temperature patterns of the forging and the die, the magnitude of the emerging stresses and deformation.

UDC 681.3.06

Principles of Industrial Software Development by Assembly Programming Methods

907G0096A Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 90 (manuscript received 29 May 89) pp 3-13

[Article by V. V. Lipayev under the rubric "The Technology and Systems of Programming": "Principles of Industrial Software Development by Assembly Programming Methods"; passages in italics as published]

[Text] **The Goals and Basic Concepts of the Industrial Technology of Assembly Programming**

The decrease of the labor intensiveness and the time of development, as well as the increase in the quality of software (PS) by the replacement of the repeated processing of similar programs with their assembly from ready-made and tested components that are developed once are the goal of the principles of the industrial technology, which are presented below [1-4]. The concept is oriented toward programming "in the large"—toward the development of integrated high-quality software with the lengthy development by collectives of specialists of the nomenclature and functional capabilities of the versions of the programs.

These principles to one degree or another originated and appeared in case of the traditional technology of designing the programs of integrated software. However, they stand out more clearly in case of assembly programming. The structuring of these principles and the distinction of three interacting technological processing (the devising of components and systems and their development) make it possible to focus attention on the methods and means, which ensure the great efficiency of assembly programming as a whole. Here the technology of the maintenance, modification, and development of the versions of components and software is regarded as a fundamental part of the technology of development.

Integrated *software,* which has successfully undergone tests and is qualifiable as a product for production engineering purposes, is the basic *objects of development* and the final products of the technology in question. In the process of development up to the completion of tests these objects are called below primarily *a program system* (KP). During designing each program system of some problem-oriented functional purpose can have several working version, which are distinguished by the level of completeness of development and by the characteristics and composition of the components. The base versions, or editions of the software, which are distinguished by the clear formalization of the characteristics and the completeness of the composition of the components and the drawing up of documents, are formed in accordance with the test results. These base versions are objects of delivery to users, on the basis of which the versions of the users of the software, which have been adapted to the specific characteristics of the application environment, are generated in accordance with the instructions set forth in the documentation.

The concept of *reusable components* (PIK) for the assembly of software envisages developed, debugged, tested, and documented components, which are suitable for application in a set of software of a specific problem-oriented purpose. Hereinafter objects, to which the data, which are to be processed, and the operations on the data, which are concentrated in the programs of their processing, correspond, are regarded as reusable components. The data are assembled into *information modules,* which have an identifier and declaration (the specification of requirements). The structure of the data can be simple or complex, with a large number of specifications of various units of data and with the characteristics of their location. The operations on data are combined into *program modules,* which are characterized by the names and specifications of the requirements, which include the functions and content of the intermodule interfaces, as well as the codes of programs at different levels of detail [5].

The program modules are combined into *groups and systems of programs,* which can also be used as reusable components. They are differentiated by the degree of processing, documentation, and accessibility for changes, as well as by categories of specialists who make the decision on their change.

The decisions on the updating of versions of the developer of the reusable components are made in the process of development, are not always documented, and are implemented on line by the immediate developer of the program or information components. The working versions of reusable components are checked and debugged within the working versions of software. The updating of reusable components is allowed with the authorization of the supervisor of the development of the working version of the software, but is carried out by the immediate developer of the reusable components, who draws up complete documentation for the checked component and the introduced change.

The process of changing *guaranteed versions of reusable components,* which have undergone testing within one of the base versions of the software and are provided with complete documentation, is most conservative and complex. In these cases the checking of the potential consequences of the updating of reusable components for the entire set of base and user versions of software and the authorization of the supervisor of the development of this set are necessary. The documented changes of reusable components or the base version of software with the appropriate comments should be known to all users.

Every software item is characterized, first of all, by functions, a structure, and an interface. For all the diversity of the functions of software their structures in many respects are similar and are based on the principles of modular hierarchical construction [6]. The standardization of the specific rules of the structured design of software within its problem-oriented areas ensures the necessary flexibility and simplicity of the modification of the base versions of software. These versions are characterized by a standardized interface of the software with the external environment of its application in the computer and in data processing and control systems, as well as in case of interaction with users. The concept of an interface reflects the formally exact specification of the composition and content of the data being obtained and issued, as well as the control signals during the functioning of the software.

The Principles of the Structured Design of Software Based on Reusable Components

In this case *modifiability* (flexibility) and the assurance of the possibility of changing the composition and functions of the reusable components and program system with the maintenance of the adopted structured design are the most important criterion. Depending on the peculiarities of the problem area the effective use of the memory or performance of the computer being realized, the labor intensiveness or time of development, and others can also be criteria when choosing the structure. In conformity with the goals and tasks of designing at the stage of systems analysis and the formation of the structure of the reusable components and software it is necessary to rank and distinguish the dominant criteria.

The effectiveness of the method of assembly programming is ensured by a number of rules of the structured design of software and its components, as well as the interaction between them. These rules are aimed at the standardization and unification of the structure and interaction of components of different ranks and purposes within the problem area of their repeated use. A certain portion of the rules is of a quite general nature and can be applied practically always. Another portion reflects the problem and machine orientation of the class of software and the set of reusable components and is subject to processing for the effective use of assembly programming in the corresponding problem area.

As a result of the systems analysis the *standardized structure of the program system* of a specific class and a *set of rules* of its formation are formed. Program modules are their basic elements. The modules can be combined into functionally complete groups of programs, which can also be used as reusable components. In such a standardized structure it is advisable to distinguish the functional program modules, which are subject to the greatest changes, the standardizable means of the checking and organization of the computing process, as well as the typical standard functions [7]. These standardized means and functions are usually connected with the peculiarities of the implementing computer and can be applied in several problem-oriented areas. At the same time they dictate a number of rules of the structured design of software and reusable components.

The program modules for reuse should be based on standardized rules of the structured design of reusable components and the drawing up of the specifications of requirements and the descriptions of codes. The basic goal of such standardization is to facilitate the development of modules and to ensure their quality, as well as to simplify the study of their functions and characteristics during reuse. These rules to a significant degree are fixed in model high-level programming languages, for example, Ada [8].

However, when using several high-level languages and assemblers it is advisable to develop additional rules of the structured design of program modules, to distinguish the typical associations of operators and the constraints of their use, as well as to introduce rules of the description of codes in the programming languages and comments, rules of the declaration of variables and headers of modules, the restriction of the size of modules and their complexity, and so on. These rules should be observed most completely when developing the bulk of functional programs that are liable to reuse in versions of software. The components of the organization of the computing process, the monitoring of functioning, and input-out and several others can have differences in the rules of structured design, which are oriented toward the peculiarities of their implementation in a specific computer and toward the peculiarities of interaction with the external environment.

For the assurance of the subsequent modification and development of versions of software it is important *to standardize the structure of the database,* in which initial, intermediate, and resultant information is accumulated and stored in the process of the operation of the software. Information modules or data batches are the basic components of this structure. In them it is also advisable to use typical structures, which are oriented toward the efficient processing of data in a specific problem area. The combining of information modules makes it possible to develop more complex structures of data of a specific special purpose. The hierarchy of the links between these components to a certain degree corresponds to the process of processing and to the flows of data and is weakly linked with the structure of the program components in the software. At the same time the deep and hard-to-control links of the information modules with the program components lead to the need for the efficient standardization of the structure and content of the database.

The *structure of information modules* is liable to unification on the basis of a number of standard versions of the location and specification of data. Several of them are restricted by the rules of the declaration of variables in programming languages, while others are developed as applied to the efficient solution of standard problems of a specific problem-oriented area. It is advisable to interconnect the structures of the information modules with allowance made for the goal and place of their use by program components. It is also desirable to distinguish the information modules which are connected with the program components that are most liable in the future to changes and modifications. Such localization of several information modules can increase the flexibility and effectiveness of the development of new software or highly modified versions.

The *unification of the structure of the intermodule interface* with respect to control transfers and with respect to information is of particular importance for assembly programming. These rules are formulated on the basis of declarations of programming languages or are drawn up on the basis of the rules of the structured design of programs and the database of the software. In the latter case the calling sequence is specified in the macrocommands of intermodule interaction. The disciplines of parameter transmission and the saving of registers when calling program modules and when returning control after a call are indicated in them. Moreover, the rules of the use of memory and registers for the storage of local variables in complex sequences of the calling of program components are specified.

The two types of program products (reusable components and software) differ substantially in the complexity of the individual object, but the overall complexity of the complete problem-oriented nomenclature of reusable components for several areas of application can significantly exceed the complexity of each base version of software. The distinction of the two types of products makes it possible in case of assembly programming to use for each of them its own level of abstraction (detailing) of the declaration of objects, as well as a different latency of superfluous information in the design language. As a result the content of the program components and the program system as a whole when developing new products can be used to a different degree [5].

There corresponds to each level of abstraction its own design language, which has common fragments with other languages. The design languages of the top levels for the development of reusable components and software can differ significantly subject to the problem-oriented areas of application.

It is advisable to examine the effectiveness of the reuse of developed reusable components and software as a whole at different levels of declaration in two aspects: the reuse of components on similar computers and the reuse of components, which were developed on some computer, with their transfer to another implementing computer, which differs in resources, the instruction set, and others. In both cases the specific characteristics of the computer, in which the program reserve is reused, can be substantial limitations for its use. These limitations appear more strongly, the more completely the intellectual and technical content of previously developed algorithms and programs is used.

The Structure and Components of the Industrial Technology of Assembly Programming

The development of integrated software in case of any technology begins with a systems analysis of the subject area of the planned application of the software and with the definition of the goal and tasks of development (Figure 1). In this portion of the technological process the prospect and effectiveness of the application and development of assembly programming in the given problem area are evaluated. If the assembly programming is quite effective, the development of the standard structure of the software, the composition of reusable components, which are necessary for the given area of application, and the general principles of memory and power allocation in case of the function of versions of the software is carried out. The subsequent detailing consists in the preparation of specifications for the basic set of reusable components and the first base versions of the software, as well as in the formulation of the rules of the specification of data, the interfaces between reusable components, and the interaction of software with the external environment.

After completion of the systems design the technological process is divided into three quite independent autonomous processes (see Figure 1):

—the development and accumulation of reusable components for their repeated application in a specific problem-oriented area of the development of software;

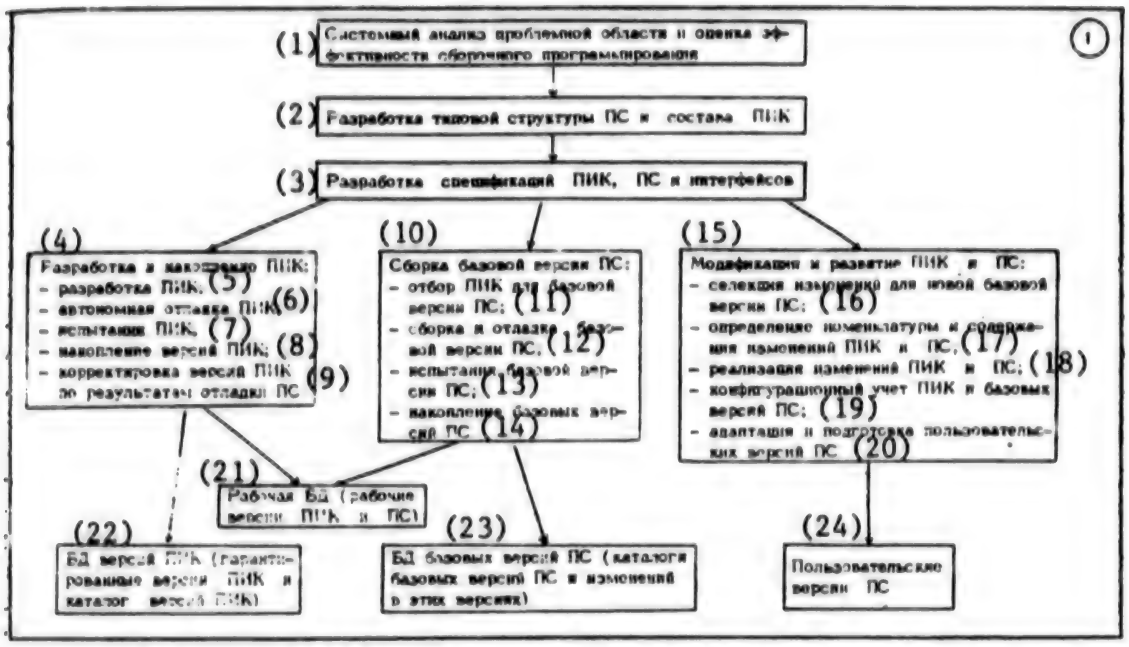
—the assembly, debugging, and testing of base versions of the software from the prepared set of program components;

—the modification and development of the composition and functions of the software and its components for the change and expansion of the characteristics of the base versions of the software.

These processes differ significantly in the technological stages and operations, the necessary means of automation and the peculiarities of their application, and the composition and skill of the specialists for their implementation. Some portion of the operations, means of automation, and specialists are involved in two and even three processes, but the basic goals and results of each of them differ significantly.

The Principles of the Development of Reusable Components

The technological process of developing reusable components is close to the traditional process [9] with the necessity of the accumulation, the assurance of the use,


1
(1) Системный анализ проблемной области и оценка эффективности сборочного программирования
(2) Разработка типовой структуры ПС и состава ПИК
(3) Разработка спецификаций ПИК, ПС и интерфейсов
(4) Разработка и накопление ПИК:
(5) - разработка ПИК;
(6) - автономная отладка ПИК;
(7) - испытания ПИК;
(8) - накопление версий ПИК;
(9) - корректировка версий ПИК по результатам отладки ПС
(10) Сборка базовой версии ПС:
(11) - отбор ПИК для базовой версии ПС;
(12) - сборка и отладка базовой версии ПС;
(13) - испытания базовой версии ПС;
(14) - накопление базовых версий ПС
(15) Модификация и развитие ПИК и ПС:
(16) - селекция изменений для новой базовой версии ПС;
(17) - определение номенклатуры и содержания изменений ПИК и ПС;
(18) - реализация изменений ПИК и ПС;
(19) - конфигурационный учет ПИК и базовых версий ПС;
(20) - адаптация и подготовка пользовательских версий ПС
(21) Рабочая БД (рабочие версии ПИК и ПС)
(22) БД версий ПИК (гарантированные версии ПИК и каталог версий ПИК)
(23) БД базовых версий ПС (каталоги базовых версий ПС и изменений в этих версиях)
(24) Пользовательские версии ПС

Three Processes Which Support Assembly Programming and the Reuse of Components

Key:—1.Systems analysis of problem area and evaluation of effectiveness of assembly programming—2.Development of standard structure of software and composition of reusable components—3.Development of specifications of reusable components, software, and interfaces—4.Development and accumulation of reusable components:—5.development of reusable components;—6.off-line debugging of reusable components;—7.testing of reusable components;—8.accumulation of versions of reusable components;—9.updating of versions of reusable components in accordance with results of debugging of reusable components—10.Assembly of base version of software:—11.selection of reusable components for base version of software;—12.assembly and debugging of base version of software:;—13.testing of base version of software;—14.accumulation of base versions of software;—15.Modification and development of reusable components and software:—16.selection of changes for new base version of software;—17.specification of nomenclature and content of changes of reusable components and software;—18.implementation of changes of reusable components and software;—19.configurational consideration of reusable components and base versions of software;—20.adaptation and preparation of user versions of software—21.Working database (working versions of reusable components and software)—22.Database of versions of reusable components (guaranteed versions of reusable components and catalog of versions of reusable components)—23.Database of base versions of software (catalogs of base versions of software and changes in these versions)—24.User versions of software

and the modification of versions of the reusable components, which are distinguished by the stages of application, the level of debugging, the characteristics of quality, and the accessibility for changes. Moreover, the reusable components acquire the status of completed components with the corresponding documentary registration and the possibility of application in various combinations. The technology of developing reusable components should be supported by means of automation, which ensure:

—the singling out of reusable components, their identification, the description of the functions and relationships;

—the development of reusable components, the testing and the assurance of the quality of the components;

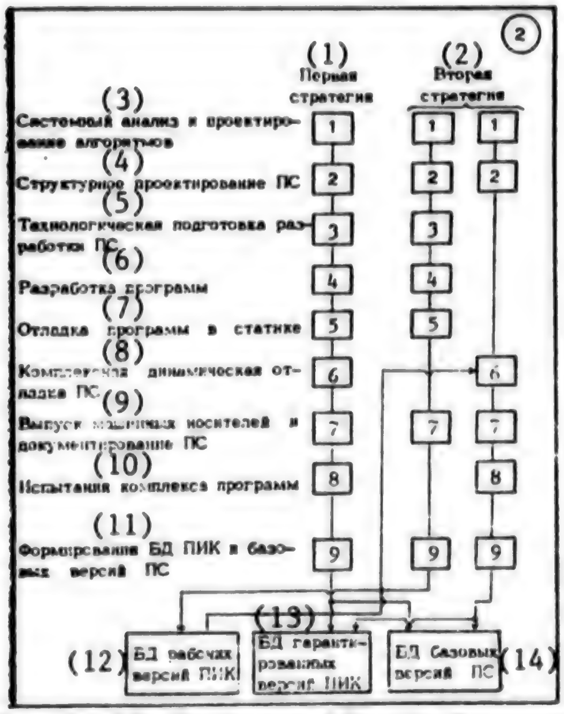
—the accumulation, ordering, and storage of reusable components;

—the retrieval of the necessary reusable components according to their functions, algorithms, and descriptions;

—the maintenance and modification of the characteristics of the existing components, as well as the development of the nomenclature of reusable components which are accessible for application.

The development and accumulation of reusable components can be based on two strategies. In case of the *first strategy* the reusable components are extracted from the tested and approved base version of the software, which has gone through the complete technological process of development (Figure 2). As a result guaranteed versions of the reusable components, which have been carefully debugged and checked within the software, are accumulated in the database. These reusable components are subsequently used directly for the assembly of new versions of the software, which are accumulated in a separate database. It should be taken into account that in the process of the writing and development of the base versions of software these sets of reusable components can change both with respect to the nomenclature and with respect to the content of each component.

The *second strategy* (see Figure 2) is oriented toward the development of a broad nomenclature of primary working versions of reusable components, which can be subsequently modified. In this case the nomenclature and specifications of the requirements for the necessary set of reusable components are determined on the basis of the systems and structured design of the problem-oriented group of software. The reusable components, which have been developed, debugged off line, and


2
(1) Первая стратегия
(2) Вторая стратегия
(3) Системный анализ и проектирование алгоритмов
(4) Структурное проектирование ПС
(5) Технологическая подготовка разработки ПС
(6) Разработка программ
(7) Отладка программ в статике
(8) Комплексная динамическая отладка ПС
(9) Выпуск машинных носителей и документирование ПС
(10) Испытания комплекса программ
(11) Формирование БД ПИК и базовых версий ПС
(12) БД рабочих версий ПИК
(13) БД гарантированных версий ПИК
(14) БД базовых версий ПС

Strategies of the Development of Reusable Components and the Optimization of Base Versions of Software

Key:—1.First strategy—2.Second strategy—3.Systems analysis and design of algorithms—4.Structured design of software—5.Technological preparation of development of software—6.Development of programs—7.Debugging of programs in statics—8.Integrated dynamic debugging of software—9.Production of machine media and documentation of software—10.Testing of program system—11.Formation of database of reusable components and base versions of software—12.Database of working versions of software—13.Database of guaranteed versions of reusable components—14.Database of base versions of software

documented, are accumulated in the database of working versions of components. Here in the process of their development the stages of integrated debugging and testing within the software are absent.

The working versions of reusable components can in a number of cases have sufficient guarantees for their conversion and use subsequently when assembling software without addition updatings. For this the possibility of transfer to the database of guaranteed versions before or after their approval within the software should be envisaged for a portion of the reusable components. Here the extent of the changes of the content of reusable components can be greater than in case of the first strategy, but the necessary nomenclature of reusable components for the entire group of software can be envisaged more completely.

A *system of the naming and identification of the functions of reusable components,* which has been standardized, at least within a class of software, is necessary for the clear understanding and description of the functions of program and information components, functional groups of programs, and structures of data. This system should ensure the complete and correct extraction of the necessary reusable components from the set of similar versions, which differ in individual characteristics, without the need for the analysis of specific algorithms, program codes, and detailed specifications of data. The program components should allow the labeling and extraction of the working versions, which are found in the process of development and debugging, the versions, for which off-line debugging with a specific reliability has been completed, and the versions of a guaranteed high quality, which have undergone testing within the software. The descriptions of the reusable components should be accompanied by the general characteristics of the quality of the solution of functional problems with respect to accuracy, the range of data being processed and put out, and so on.

The *descriptions of the specifications of requirements* should contain a hierarchically ordered set of characteristics of the components, which ensures the cataloging of reusable components, the automatic retrieval of a group of the most suitable ones of them in accordance with their descriptions, and the final selection of components with allowance made for the entire nomenclature of functional and technical indicators. The availability of prepared source codes of programs and the language of their description and the availability and location of compiled object and load modules for a specific type of implementing computer are indicated within the characteristics that are stored in the database of the design of software. The hierarchy of characteristics of reusable components is necessary for the effective use of descriptions at the different stages of their development and application when writing software.

When developing reusable components the *rules of the structured design of program components,* which ensure their potential correctness and debuggability, should be followed. The recommendations and restrictions on the use of the statements of the programming language, on the standard structures of groups of statements, on the size and complexity of program modules, on the use of declarations of local and global variables, and so forth are recorded in these rules. The rules of the checking of the functioning of program components and protection against the erasing of data are formalized.

The *information modules* are broken down by levels of accessibility from the program components and should have descriptions, which are standardized in the form and structure of the contents. It is advisable to use unified descriptions of the standard structures of data, which are formed into information reusable components.

With allowance made for the problem-oriented area of the application of software, the characteristics of the computer, and the operating environment of functioning a standardized *system of rules of intermodule interfaces* should be used. These rules take into account all the peculiarities of the programming language, the characteristics of the operating systems being used, and the structures of the software, as well as the possibilities of the development and modification of versions of the program components. The standard statements of the interaction of components should ensure the flexibility of the change of the structure of the software and immunity to errors in case of such changes.

Complete documentation, which ensures the application and the complete and clear understanding of the characteristics of reusable components, is drawn up for the program and information components. In the documentation it is necessary to stipulate a hierarchical structure of descriptions, which gradually expands from the level of the designation and brief specification of the requirements to the texts of programs in object code and the set of tests, on which the component was checked. The structure, content, and method of naming of the documents for reusable components should allow the making up by them of the complete body of operating and technological documentation for different versions of the software.

The Principles of the Assembly of Software From Reusable Components

The technology of assembling software (see Figure 1) should ensure the stage-by-stage development of versions of program systems at different levels of description of the components. In conformity with the levels of representation of the reusable components the software versions being developed differ in purpose, quality, and readiness for application in a real environment. The successive optimization of the versions of software at different levels of description ensures the effective implementation of the stages of design from the systems analysis of the posed task to the completion of the testing of the finished base version of the software of desired quality. A new base version of the software can be assembled from components of different size and functional complexity.

In the technology of assembling software it is necessary to envisage *standard variants of the state of the nomenclature* of reusable components that are ready to use:

—the practically complete absence of developed reusable components when developing the first prototype of the software, from which the bulk of the reusable components for subsequent versions should be extracted;

—the availability of a large number of reusable components, which are necessary for the assembly of the next version of the software, when it is necessary to develop only a portion of the new components that will supplement the existing nomenclature;

—the availability of all the prepared and approved reusable components, which are necessary for the assembly of the desired version of the software.

In the first variant of the technology the development of reusable components and the assembly of software are combined in a single technological process. It differs from the traditional process [9] only by the singling out, cataloging, and accumulation for subsequent application of the reusable components, which have undergone debugging and testing within the first base prototype of the software (see Figure 2). Here all the rules of the optimization of the reusable components, which are suitable for assembly in subsequent versions of the software, should be observed.

In the second variant, when the majority of reusable components, which have been debugged within the base versions of the software, have been developed and are present in the design database, their assembly in a new version of the software is the basis of the technology. The necessity of developing several new reusable components leads to the partial use of the technology of developing components. The availability of the bulk of optimized reusable components can significantly facilitate and modify the process of developing the necessary new reusable components. Here their debugging in a number of cases can be carried out directly in the software version being integrated. This is especially effective when the new reusable components are not concentrated in a group, which solves a new functional problem, but are distributed among one group or small groups in various functional components of the software.

In the third variant the need for the development of new reusable components is absent and the assembly technology is implemented in *most complete form.* The peculiarities of the new base version of the software can cause the need for the updating of several reusable components. These changes are subsequently carried over to all the previously developed versions of the software, in which updated reusable components are used. The technology of the development, cataloging, and selection of versions of reusable components, which are close in functional purpose and specifications of requirements, is an essential part of the technology of assembling software.

A *developing design database* of program and information components, which have been approved and are suitable for repeated use, is the basis of the technology of assembly programming. The design database is supplemented as needed with new optimized reusable components or updated versions of old components, with expanded or modified functions. Base versions of the software are accumulated and stored in individual sections of the design database. These versions are registered together with the tests and test results, as well as with the complete documentation, which is necessary for application and maintenance.

For the implementation of the technology of assembly programming the software should comply with the precise rules of the modular hierarchical design of program groups and systems. The hierarchical structure of program systems supports the technology of designing integrated software according to the principle from the top down from the standpoint of the consideration of the functional purpose and the best solution of the target problem of the entire system. This guarantees the *conceptual unity of algorithms and programs* and the possibility of the efficient distribution of limited resources of design as the components decompose.

Assembly from the top down begins with the modules of the software, which were made up from the specifications for major components, and is developed by their subsequent detailing, up to specifications and texts of the program and information modules or previously developed parts of the software, which are selected from the design database. Given a complete nomenclature of program codes and data specifications the selected reusable components are made up into completed program groups also from the top down, starting with the operating system. The possible gaps in the necessary composition of ready-made reusable components can be temporarily filled by "plugs"—specifications or simplified models of components. The multilevel hierarchical design of software makes it possible to limit and localize at each of the levels the components corresponding to it, as a result of which the analysis and synthesis of components of higher levels and the system as a whole are facilitated.

The technology of assembling software on the basis of reusable components should ensure the *checking of the control and information links* between components of the software. When assembling the software it is necessary to establish the identity of the specific control interactions in each pair of calling and called program modules. Each information link is taken into account in the modules, which have some variable at input, as well as in the modules, which have the same variable at output. When making up reusable components and debugging software the identity of each information link should be checked and their identical nature should be ensured.

Although the technological processes of developing reusable components and software are separate, interaction, which it is necessary to take into account, exists between them. Let us note, first of all, the *peculiarities of the testing and debugging* of program components and software as a whole. The relative simplicity of reusable components makes it possible to use most completely deterministic testing, to plan the sequence of the application of tests, and to evaluate the completeness of the debugging of components. However, it is not always possible to guarantee a high level of debugging of reusable components before the checking of their functioning within software. It is practically impossible to carry out the off-line debugging of reusable components on a real-time scale and in interaction with the external environment.

The great complexity of software as compared with its integral components reduces the possibility of using deterministic tests in case of limited resources for debugging. As a consequence the testing and debugging of software are distinguished by the use of primarily statistical tests and in necessary instances real-time tests. This complicates the reliable evaluate of the completeness of the testing and the quality of software as a whole, but at the same time increases the level of debugging of reusable components.

The assembly of the base version of software concludes with the *integrated debugging and testing* of programs in a real external environment or in case of its simulation. For this it is necessary to develop models of the external environment, which are capable of generating deterministic and stochastic test data, as well as tests on a real-time scale for the corresponding class of software. The models of the external environment, which were used during the testing of the software, the initial data in case of the generation of the base set of tests, as well as the resultant data are cataloged and accumulated in the database together with the prepared base version of the software. The changes of the tests and test results should correspond to the changes in the process of maintaining the base versions of the software.

There is drawn up for each base version of software a *set of operating documents*, which ensure its delivery to the user, skillful use, and the generation of versions of the users. These versions are created in conformity with the conditions of the specific applications environment by the adaptation of the base versions of the software in accordance with the rules that are presented in their operating documents. The developers accumulate *technological documentation* for the base versions, which ensures their maintenance, development, and modification.

The Principles of the Modification and Development of Reusable Components and Software

Computer software is one of the most flexible types of items, which are subject to changes and development over their entire life cycle. Here a significant portion of the changes fall to the period of the use and maintenance of programs. Therefore, a technological process, which

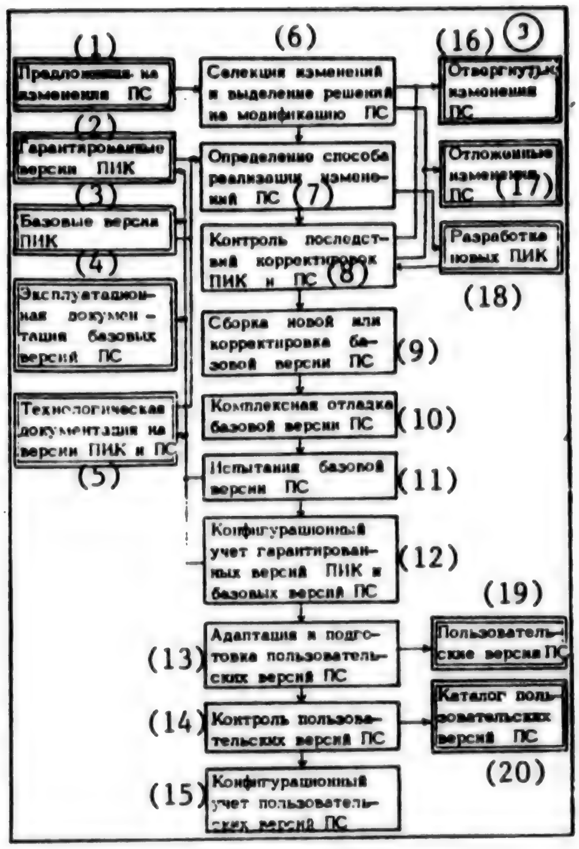

(1)
Предложения на изменения ПС
(2)
Гарантированные версии ПИК
(3)
Базовые версии ПИК
(4)
Эксплуатационная документация базовых версий ПС
Технологическая документация на версии ПИК и ПС
(5)
(6)
Селекция изменений и выделение решений на модификацию ПС
Определение способа реализации изменений ПС
(7)
Контроль последствий корректировок ПИК и ПС
(8)
Сборка новой или корректировка базовой версии ПС
(9)
Комплексная отладка базовой версии ПС
(10)
Испытания базовой версии ПС
(11)
Конфигурационный учет гарантированных версий ПИК и базовых версий ПС
(12)
Адаптация и подготовка пользовательских версий ПС
(13)
Контроль пользовательских версий ПС
(14)
Конфигурационный учет пользовательских версий ПС
(15)
(16)
3
Отвергнутые изменения ПС
Отложенные изменения ПС
(17)
Разработка новых ПИК
(18)
(19)
Пользовательские версии ПС
Каталог пользовательских версий ПС
(20)

Diagram of the Modification and Development of Base Versions of Software

Key:—1.Proposals on changes of software—2.Guaranteed versions of reusable components—3.Base versions of reusable components—4.Operating documentation of base versions of software—5.Technological documentation for versions of reusable components and software—6.Selection of changes and choice of decisions for modification of software—7.Determination of method of implementing changes of software—8.Checking of consequences of updatings of reusable components and software—9.Assembly of new version or updating of base version of software—10.Integrated debugging of base version of software—11.Testing of base version of software—12.Configurational consideration of guaranteed versions of reusable components and base versions of software—13.Adaptation and preparation of user versions of software—14.Checking of user versions of software—15.Configurational consideration of user versions of software—16.Rejected changes of software—17.Postponed changes of software—18.Development of new reusable components—19.User versions of software—20.Catalog of user versions of software

ensures the *ordered and logged making of the necessary changes* with the simultaneous guaranteed improvement of the quality of software, is necessary. The technology of modifying and developing the base versions of software and their integral components is specific in assembly programming. It affords the possibility of the prompt and long-term, controlled change of program components and software with the broadening and modification of the demands on them, as well as for the increase of the quality of functioning (Figure 3).

Changes and expansions of the characteristics of the external environment of the functioning of software, the proposals of users and clients on the broadening of the functions and the improvement of the quality of software, and errors, which were identified in the process of use or testing, are the *sources and causes* of possible changes of programs.

The listed stimuli for changes of programs should be specified, formalized, and accumulated in the database together with the characteristics of the conditions and sources of their appearance. The technology of developing programs should ensure the *analysis and evaluation of possible changes of programs* in accordance with the following indicators:

—how much the given change can expand and improve the operating characteristics of the software as a whole;

—what the possible expenditures on the making of updates and their dissemination of users are;

—for what number of users the given change might be useful and what the urgency of its making is;

—how the given change might affect the use of previous versions by users.

The evaluations of proposals for changes of programs serve as the base for their *selection and the preparation of decisions for the modification* of software and components. The technology of developing programs should ensure the breakdown of possible changes into groups:

—urgent changes, which should be reported to users for the prompt updating of programs before the introduction of the official base version of the software;

—changes, which it is advisable to make in the next base version, since they justify the expenditures on their implementation and the improvement of the quality of the software;

—changes, which require additional analysis of the advisability and effectiveness of their implementation in subsequent versions and might not be introduced in the next version;

—changes, which do not justify expenditures on development and the making of updates or practically do not influence the quality of the software and, therefore, are not liable to implementation.

The changes, which have been singled out for implementation, are subject to *evaluation and the determination of* *the method of their fulfillment.* The technology of developing base versions of software should envisage:

—the assembly of a new base version of the software with a changed structure and composition of reusable components given the significant change of the functions of the software;

—the addition to the existing base version of the software of new reusable components from those that are ready and are available in the database;

—the development of new reusable components for the implementation of previously not envisaged functions in the existing base version of the software;

—the updating of several reusable components for the improvement of the characteristics of the base version of the software with the subsequent development of additional versions of the reusable components or with the carrying over to the changed reusable components to the previously developed versions of the software.

The technology of implementing the changes should ensure the *checking of the immediate and remote consequences of updatings* of reusable components and the software as a whole. It is necessary to carry out such checking from the top down—from the general functions and specification for the software, and from the bottom up—from the specific change of the code of the program of reusable components to the characteristics of the software. Since even small changes of individual reusable components can affect the most important functional characteristics of integrated software, in case of collective development the *regulation of updatings and the discipline of their fulfillment* are of particular importance. For this each change, which has been singled out for implementation, is checked in succession in the program or information module, in the group of functionally linked programs, in the working version, in the prototype, and in the base version of the software. At each level of debugging and checking the corresponding specialists authorize and make the changes, moreover, among these specialists the level of responsibility for the components and versions of the software increases in succession.

It is necessary to support the processes of changing reusable components and software by the *methods and means of configuration management,* which ensures the analysis, consideration, and registration of all versions of programs and the changes made in them. A special language of design and description of updatings, as well as means of the automation of the technological process are characteristic of this technology, just as of other technologies.

The technological process of the analysis, selection, implementation, checking, and dissemination of changes of integrated software relies on a *developed database of program components and software,* which are at different stages of development. The database is subject to careful protection against unauthorized access and against random erasing. The components accumulated in them are accompanied by the necessary documentation and tests, which ensure the complete rechecking of the quality of functioning. The assurance of the complete mutual correspondence of the source codes of programs and data, the object and load modules, the content of the documentation, and the texts of the specific base version of the software on data media is of great importance.

These data should be selected and generalized in the technological documentation of the development and maintenance of versions of software of a specific problem orientation. The configuration management and checking of the periodic release of optimized and tested base versions of software are carried out on the basis of these data. The data on changes and the development of new base versions should be reported to users for the making by them of decisions on switching to the new versions.

The set of listed data should ensure the continuous increase of the quality and the broadening of the nomenclature and function of software on the basis of reusable components with the simultaneous decrease of the specific labor intensiveness of their development.

Bibliography

1. A.P. Yershov, "An Attempt at an Integrated Approach to the Urgent Problems of Software," KIBERNETIKA, No 3, 1984, pp 11-21.

2. C.V. Ramamoorthy, V. Garg, A. Prakash, "Support for Reusability in Genesis," IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, Vol 14, No 8, 1988, pp 1145-1154.

3. R. Prieto-Diaz, R. Freeman, "Classifying Software for Reusability," IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, No 1, 1987, pp 6-16.

4. V.P. Kotlyarov, "The Technology of Developing the Software of Built-In Microcomputers and Its Supporting Development Complexes," MIKROPROTSESSORNYYE SREDSTVA I SISTEMY, No 5, 1987, pp 29-35.

5. V.V. Lipayev, L.A. Serebrovskiy, "The Evaluation of the Effectiveness of the Development of Portable (Mobile) Programs," UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2, 1987, pp 3-8.

6. V.V. Lipayev, V.V. Filippovich, "The Principles and Rules of the Modular Design of Integrated Program Systems of Automated Management Systems," UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 1, 1975, pp 15-22.

7. V.V. Lipayev, "Testirovaniye programm" [The Testing of Programs], Moscow, Radio i svyaz, 1986, 292 pages.

8. R. Bar, "The Ada Language in System Design," Moscow, Mir, 1980, 320 pages.

9. V.V. Lipayev, L.A. Serebrovskiy, P.G. Gaganov, et al., "Tekhnologiya proyektirovaniya kompleksov programm ASU" [The Technology of Designing Program Systems of Automated Management Systems], Moscow, Radio i svyaz, 1983, 263 pages.

UDC 681.325

Design and Generation of Software of Flexibly Computerized Manufacturing Systems on the Basis of Standardized Modules

907G0096B Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 90 (manuscript received 8 Dec 88) pp 75-81

[Article by V. G. Tsyvinskiy, D. A. Rossoshinskiy, and N. V. Tkachenko under the rubric "Integrated Data Processing and Control Systems": "Design and Generation of Software of Flexible Computerized Manufacturing Systems on the Basis of Standardized Modules"; passages in italics as published]

[Text] Flexible manufacturing systems at present are one of the basic directions of the increase of the technological feasibility of production. A large number of publications are devoted to the study and development of lines, sections, and systems. Here the basic attention [1-6] is devoted to problem issues: the determination of the configuration of the system, the construction of computer-aided design systems of lines, the number and performance of individual modules, and so forth.

It is proposed to use general-purpose operating system software like RAFOS, OS RV, and MIOS as the run-time systems in processing nodes based on computers like the Elektronika-60, the MS1211, the SM-4, and other computers. Here it is necessary each time to achieve all over again the synchronization of the processes, which organize the operation of individual modules, and the display of the status of processes, by using the set of primitives of the chosen operating system and introducing databases (DB's) for the display of the status. This task becomes complicated in case of the use of distributed systems of automation, in which specific netware is needed.

In this work the construction of a run-time system for processing nodes of flexible sections and lines is examined. The run-time system has a higher logical level, which fixes specific assumptions with respect to the structure of systems of automation.

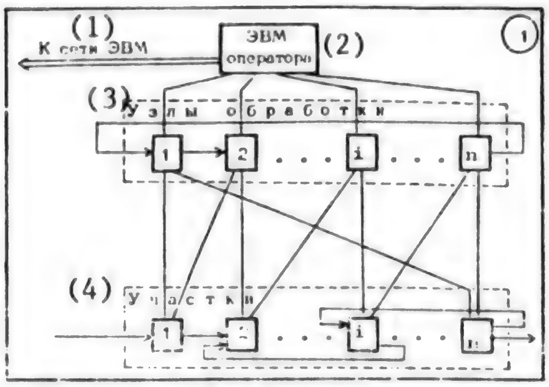
The Structure of the Line and the System of Automation

A line is a series of sections, which are represented, in turn, by directed graphs, the nodes of which are modules of the processing of objects of the given production process, while the edges indicate the direction of the flows of these objects. Depending on the systems functions being performed, the modules can be assigned to some type from a certain functionally complete set.

A system is a series of processing nodes, which are incorporated in the line. Each node, for the most part, corresponds to one section and is connected with the left and right adjacent nodes, as well as with the central (console) computer. The extreme nodes can be interconnected, forming a ring. In such a structure the introduction of a simple discipline of the redundancy of nodes: "the right one replaces the left one," is possible. Here each node has a connection with the equipment of its own section and the left section (Figure 1). There corresponds to the module, which is assigned to some section, the process in the processing node, which is connected with this section. The object of processing is represented in the system of automation by the specifier of the object (the descriptor).

As a whole the structure of the system of automation does not impose rigid restrictions on the "sections—processing nodes" relation. The basic requirement for the determination of the structure is that the processing node should not decrease the performance of the line in any appreciable manner. It is also desirable to assign the group of sections with cross-connections to one processing node, in order to reduce the through traffic of messages through the nodes.

The increase of the logical level of the system at the stages of the preparation and execution of programs is achieved by the introduction of the standardization of modules. A parametrized procedure, which describes all the actions that are connected with the synchronization of the process of module control in a query language, corresponds to each type of module. The queries are implemented by a reenterable procedure—Interpreter. The type of module also determines the content of the elements of the static and dynamic databases, which are assigned to each module. Here a high flexibility of the software is achieved. The change of the topology of a



The Structure of the Production Line and the System of Automation

Key:—1.To the computer network—2.Computer of operator—3.Processing nodes—4.Sections section (line) takes place only in the databases; changes in equipment (for example, the change of the actuating mechanism) leads to the change of the only procedure that is connected with this mechanism. The set of queries of Interpreter, which realizes the interaction of the processes that reflect the module, is functionally complete for the assurance of the openness of the run-time system and the possibility of the inclusion in it of new procedures which correspond to other types of modules.

The Diagram of Operation

There is cited in Figure 2 the diagram of the operation of the run-time system of the processing node, in which it is possible to distinguish three execution environments:

—the real-time process control operating system (OSUP RV);

—the run-time system of the language processor;

—the Interpreter reenterable program of the implementation of queries.

The process control system ensures the synchronization of processes and makes various means of interaction available to them and directly supports the implementation of the processes of communication with adjacent processing nodes and the central (console) computer, as well as other possible systems processes, for example, the processing of statistical data, output to the printer, and so forth.

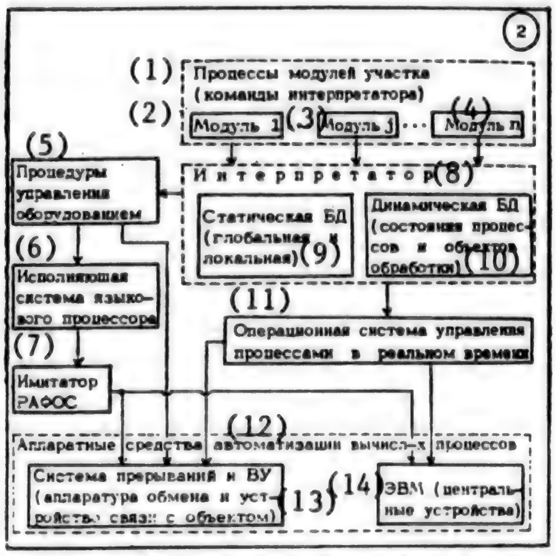

2
(1) Процессы модулей участка (команды интерпретатора)
(2) Модуль 1
(3) Модуль j
(4) Модуль n
(5) Процедуры управления оборудованием
(6) Исполняющая система языкового процессора
(7) Имитатор РАФОС
(8) Интерпретатор
Статическая БД (глобальная и локальная) (9)
Динамическая БД (состояния процессов и объектов обработки) (10)
(11) Операционная система управления процессами в реальном времени
(12) Аппаратные средства автоматизации вычисл-х процессов
Система прерываний и ВУ (аппаратура обмена и устройства связи с объектом) (13)
(14) ЭВМ (центральные устройства)

Diagram of the Operation of Machining Node *i*

Key:—1.Processes of modules of section (instructions of interpreter)—2.Module 1—3.Module j—4.Module n—5.Equipment control procedures—6.Run-time system of language processor—7.RAFOS simulator—8.Interpreter—9.Static database (global and local)—10.Dynamic database (statuses of processes and objects of processing)—11.Real-time process control operating system—12.Hardware of the automation of computing processes—13.Interrupt system and external devices (interchange and device of communication with object)—14.Computer (central devices)

The run-time system of the language processor supports the implementation of procedures of the control of the equipment of the module (actuating mechanisms, sensors, alarms, and so forth), which can be prepared within the RAFOS operating system and in Pascal, Fortran, C, and macroassembler. The run-time system is augmented by a RAFOS simulator, which implements a number of systems queries of RAFOS. The procedures of equipment control and communication with other processing nodes have access to the interrupt system and the registers of the interfaces of external devices (communication with the object and intercomputer communication).

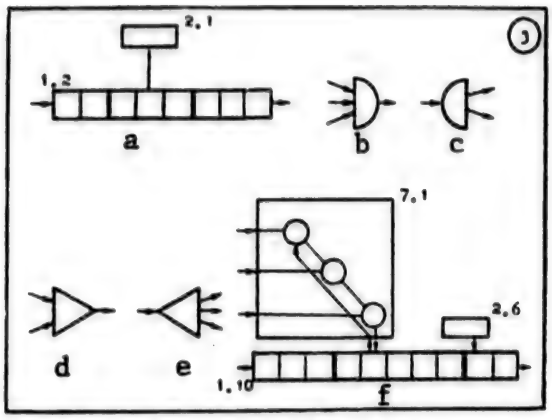
Interpreter organizes the execution of the processes, which are connected with the modules, making available to them the necessary means of interaction of the real-time process control operating system in conformity with the type of module and the topology of communication among modules. It also ensures the single-step (debugging) mode of operation and the maintenance of the database, which displays the operation of the section and the movement of the objects of processing (the dynamic database). The static database displays the configuration of the section and includes a description of its topology (the global portion) and the parameters (the local portion).

Types of Modules

The systems analysis of the structure and operation of flexible sections and lines with respect to a number of sectors makes it possible to distinguish at present seven types of modules (this number may increase as experience of designing is gained), which are typical of a broad class of discrete production processes.

The *transport module* moves discretely the objects of processing within a section (line) and connects all the modules. Each transport module is characterized by input and output modules, which are hooked up to it, by the number of cells, which contain objects of processing, as well as the number of cells in its input and output bins (storages), which play the role of buffers in case of a different performance of the modules. Without the loss of generality it is possible to consider that technological operations are performed on the objects, which are in the cells of the transport module, while the cells of the bins of the transport modules are the inputs and outputs of modules not of the transport type. Transport module 1.2, which consists of eight cells (two cells in the input bin and three in the output bin), to the second cell of which technological module 2.1 is hooked up, is shown in Figure 3, a.

The *technological module* ensures the performance of a number of operations on the object of processing, which change its state. The object is located in one of the



Graphical Designations of Types of Modules

working cells of the transport module, which is connected on the graph of the section by the "input-output" edge with the given technological module.

The *merging module* (Figure 3, b) is intended for the combining of two or more streams of processing into one. The module is characterized by the number of inputs, which are hooked up to transport modules.

The *branching module* (Figure 3, c) ensures the division of the stream of objects of processing into two or more streams, that is, performs actions that are the "reverse" of the actions of the merging module. Here the object is removed from one transport module and is placed in some other one. This module is characterized by a specific number of outputs, which are hooked up to various transport modules.

The *assembly module* (Figure 3, d) transforms objects of processing, for example, it assembles from two nodes a united object of processing. In much the same way as the merging module it is characterized by an output and a specific number of inputs which are hooked up to a transport module.

The *development module* (Figure 3, e) is "reverse" with respect to the assembly module and is characterized by one input and a specific number of outputs which are hooked up to a transport module.

The *integrated assembly module* makes it possible to perform on the object several assembly operations simultaneously. The module is characterized by one output, which is hooked up to a transport module, in the cell of which the result of assembly is placed (in contrast to the module of "conventional" assembly, the result of which is placed in the input bin of the transport module), as well as by a specific number of inputs which are hooked up to a transport module. Integrated assembly module 7.1 with three inputs and an output, which is hooked up to the second cell of transport module 1.10, is shown in Figure 3, f. Technological module 2.6 is hooked up to the third cell of the transport module.

As was indicated above, each module of a section is represented by a process, the context of which includes a procedure, which is determined by the type of module, while the specifiers (descriptors) of the objects are a representation of the objects of processing. Each process depending on the module connected with it performs specific actions on the descriptors.

Thus, the process, which represents the transport module, displaces the pointers to the specifiers of the objects of processing. The technological module is represented in the process, which changes the state of the object of processing and accordingly registers the change of state in the specifier. The representation of the activity of the merging and branching module in main memory consists in the relocation of the references to the specifiers of objects from one array of record to another. The processes, which form new specifiers of objects, are the representation of the assembly and disassembly modules.

Interpreter. Interpreter is one of the components of the virtual machine, which is intended for the elaboration of modules, while memory, which is represented by the static and dynamic databases (see Figure 2) with the accordingly specified operations of access, is the second component. Memory management is supported by the real-time process control operating system.

The static database is defined during the translation of the system definition and contains the instructions of the interpreter, as well as all the constant information on the relative position of the modules on the line and on their characteristics. There is formed for each module its own instruction sequence, moreover, the topology of the network has a significant influence on its formation. The dynamic database contains tables of queries to the kernel of the real-time process control operating system, counters, process status flags, within which the execution of a specific module is organized, buffers for the input and output of messages, synchronization semaphores, and references to the descriptors of specific objects, the operations on which are performed in automated sections.

The instruction set of the interpreter contains 23 instructions, which it is possible to divide into the following groups: the synchronization and interaction of modules, the transfer of control to subroutines, as well as reset-control.

The first group of instructions is implemented by means of the P and V operations on semaphores and by the sending-receiving of messages in accordance with the password of the real-time process control operating system. In all there are 11 instructions of this type. There is one instruction of the transfer of control to processing subroutines. It implements an unconditional jump to the procedure of the control of equipment (for example, the actuating mechanism of a robot), which is user-programmable. The group of reset-control instructions realizes several special functions, for example, the selection of the operand, the resetting of the counter of the bits, which register the movement of objects of processing on the transfer belt, and others. Several of the instructions for the listed groups are presented in the table.

Instructions of Interpreter

Mnemonics	Action Performed
PB^{∞}	The receiving of the message is unconditional, that is, it is repeated (time approaches ∞), until the query is fulfilled; an individual (IPVkh) or general (OPVKh) password can be the operands; the instruction is used for the synchronization of the operation of a module of the given type as viewed from the input
$PB^{\infty}D$	The instruction is analogous to the preceding one, but the password is formed dynamically in accordance with the result of the execution of the preceding instruction <PB^{∞}, OPVkh>; the instruction is used for the processing of the interaction of modules as viewed from the input, if by some moment of time it is not known in advance which of the modules as viewed from the input will be ready for operation
PSB	The sending of the message is unconditional and is performed with an individual password on output (the operand is IPVykh); the instruction is used for the synchronization of the operation of the module as viewed from the output
V00, P00, V10, P10	Conventional V and P operations at semaphores (input—00 or output—10); the instruction is used in a number of cases for the synchronization of the operation of modules as viewed from inputs and outputs
BPA	Unconditional address jump to the procedure of processing; the procedure is specified by the user and in each specific case can depend both on the type of module and on the section and technological process, in which the module of the given type is used
S	"Shift of belt"; the marking of the state of the transfer belt (the presence and position of items at the corresponding sites or cells for the transport module) is carried out
BPVBK	Unconditional jump to the channel selection procedure; the instruction is used for the specification of the channel that is ready for interaction as viewed from the output for the branching module

Let us examine the programs of individual modules.

For the module of the technological type at the stage of the generation of software the following instruction sequence is formed: $<PB^{\infty}, IPVkh>_i$, $<PB^{\infty}, IPVkh>_i$, $<BPA, PROG<_i$, where PB^{∞} and BPA are the mnemonics of the instructions, the meaning of which is cited in the table; IPVkh and PROG are the operands of the instructions, which specify respectively the individual password on input and the address of the processing program; the index *i* next to the right angular bracket signifies the number of repetitions of the instruction.

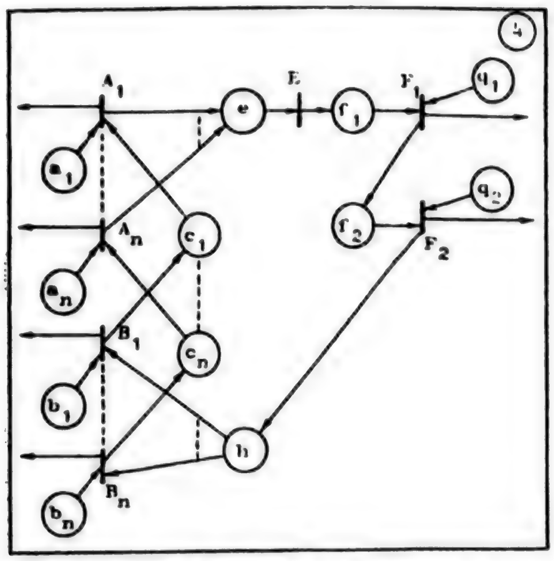
The first instruction PB^{∞}, which corresponds to the receiving of the message, implements "feedback" of the technological module. This instruction makes it possible to synchronize the operation of the module as viewed from the input (with respect to the given technological module), that is, it reports that this module is ready for operation (or has completed the preceding operation). The second instruction confirms the start of the execution of the processing procedure, which implements some technological process. After the completion of the third instruction the sequence is cycled.

During the generation of the system a sequence of instructions is formed for the given type of module. In particular, the number of repetitions of each instruction (*i*) and the specific values of the operands: IPVkh and PROG, are specified. So that the interaction on input and output would be accomplished in precise conformity with the topology of the network, individual passwords are formed as the concatenation of two bytes: in the high byte—the code of the process-source (from 1 to 256) and in the low byte—the code of the process-receiver (from 1 to 256). The process-receiver and the process-source are specified at the moment of interaction by the message exchange facilities of the real-time process control operating system [7]. From the standpoint of the user's view these are modules of different types, which are ordered by the relationship: "in the network there is a module X, the output of which at hooked up to the input of some module Y."

It should be noted that in some cases for the instructions, which implement the interaction of processes by means of message exchange, a general password, which is determined by the concatenation of the code of zero in the high byte and the code of the process-receiver in the low byte, is formed. Such passwords are formed during the translation of the instructions which specify the operation of the merging modules. As viewed from the input in the network several other modules, possibly of different types, are hooked up to such modules. From the standpoint of operation the process, which corresponds to the description of the work of the given type of module, should carry out the receiving of the message of any of the processes, which implement the control of the modules that are hooked up to the given module as views from the input. For the merging module the following instruction sequence is formed: $<PSB, IPVykh>_i$, $<PB^{\infty}, OPVkh>_i$, $<PB^{\infty}D, PAR>$, $<BPA, PROG>_i$, $<PSB, IPVykh>_i$, where PSB, $PB^{\infty}D$, and PSB are the mnemonics of the instructions (see the table), OPVkh is the general password on input, and PAR is the reference to the cell (in the dynamic area) for the formation of the password in conformity with the result of the processing of the preceding instruction.

It is possible to illustrate the execution of the instruction sequence by the Petri net in Figure 4, in which the following positions and jumps exist [8].

Positions: a_1-a_n—the modules of the inputs (n-inputs) have completed their operation; b_1-b_n—the modules as viewed from the input await the readiness for operation of the merging module; c_1-c_n—the merging module awaits the readiness for operation of modules as viewed from the input; e—the performance of the work of the



Petri Net Which Simulates the Operation of the Merging Module

merging module (the uniting of streams of items); f_1—the notification of the module as viewed from the output of the fact that the procedure of uniting has been executed; f_2—the specification of the readiness of the module as viewed from the output for the following cycle of work; g_1—the module as viewed from the output has completed the preceding operation and is ready for a new cycle; g_2—the module as viewed from the output has completed the current operation and is ready for the next cycle; h—the waiting by the merging module for the readiness for operation of one of the modules as viewed from the input.

Jumps: A_1-A_n and B_1-B_n—the performance of the interactions which coordinate the operation of the modules as viewed from the input; E—the completion of the uniting of streams of items; F_1—the module as viewed from the output begins its cycle of operations; F_2—the module as viewed from the output completes the operation.

The initial position of the token in the presented model is in f_2. The execution and completion of the instructions PSB correspond in the model to positions f_1, f_2, g_1, and g_2 and to jumps F_1 and F_2. For the instruction BPA these are respectively position e and jump E. The part of the model c_1-c_n, a_1-a_n, and A_1-A_n corresponds to the instruction $PB^{\infty}D$, that is, in this case the receiving of the message from a specific process, which corresponds to some chosen module as viewed from the input, is carried out. The dynamically formed password is located in the cell PAR. The formation of the password is determined by the result of the execution of the preceding instruction PB^{∞}. Positions and jumps h, b_1-b_n, and B_1-B_n correspond to this instruction. The interpreter instruction PB^{∞} is simulated by the query for the receiving of a message in accordance with the password. When the message is being received, the query table contains the code of the process-source and, thus, all the necessary information is available for the formation of an individual password (for the next instruction $PB^{\infty}D$).

Implementation

The real-time process control operating system, which was implemented for microcomputers of the Elektronika-60 family, was chosen as the operating system for the support of the execution of the basic operations of Interpreter—the synchronization and interaction of processes, memory and data management, and others [9]. The control program of the operating system is resident in the main memory of the microcomputer; the possibility of the location of the control program and the functional programs of the user in permanent or semipermanent memory is ensured. At the same time the mode of execution in the environment of the RAFOS (FODOS) operating system is one of the modes, for example, in the process of debugging the prototype system. The control program supports the execution of programs in macroassembler and Pascal.

The control program of the real-time process control operating system implements the following basic means for the organization of the real-time execution of functional programs: dynamic priority supervision, the synchronization of processes by P and V operations, memory allocation in accordance with queries, the interaction of processes on the basis of the mechanism of the receiving-sending of messages (this protocol of the interaction of processes at the applications level is implemented in case of the uniting of several microcomputers into a local area network), the periodic starting of processes and the checking of their execution in accordance with the time out (according to the "pure" processor time or the total time of execution).

At the level of the interaction of processes for the implementation of the approach described in the article the direct exchange of messages among processes with the use of passwords is used [10]. The accomplishment of exchange is realized by two operations: POSLAT (Pr, Ms) and PRINYAT (Ps, Mr, T), where Ps and Pr are identifiers, substitutes of identifiers, or passwords of processes; Ms and Mr are buffers for the location of data in the process-source and the process-receiver; T is the time slice, which has been allotted to the process-receiver for the operation PRINYAT.

When using passwords (16-bit binary code) all the processes, which have the necessary password and have executed the operation PRINYAT, until the moment of the processing of the instruction POSLAT are located in the queue of processes that are waiting for messages. If for a process, which issued the query PRINYAT, there is no message, it is blocked until the receipt of a message or until the passage of time T, which has been allotted to it for the execution of the operation PRINYAT. Until a message is received, the process, which executes the operation POSLAT, is in the waiting phase (is halted) in a specially organized queue of processes that send messages.

The generation of the components of the run-time system is carried out by the processing of the system structure file, which includes information on the connections between sections and the structure of the sections, that is, on the modules, which are included in the section, and the connections between nodes. The subfile, which pertains to one node, contains information on the number of modules of each time and descriptions of the specific characteristics of each module. At present such a file is prepared manually. For the development of the system it is necessary to supplement it with interactive means of the definition of the configuration of the line and sections and with special programs of the preparation of automatic monitoring and analysis.

Bibliography

1. "Gibkiye proizvodstvennyye kompleksy" [Flexible Production Complexes], edited by P.N. Belyanin and V.A. Leshchenko, Moscow, Mashinostroyeniye, 1984, 384 pages.

2. Yu.V. Ivanov and N.A. Lakota, "Gibkaya avtomatizatsiya proizvodstva REA s primeneniyem mikroprotsessorov i robotov" [The Flexible Automation of the Production of Electronic Equipment With the Use of Microprocessors and Robots], Moscow, Radio i svyaz, 1987, 464 pages.

3. A.A. Ivanov, "Gibkiye proizvodstvennyye sistemy v priborostroyenii" [Flexible Manufacturing Systems in Instrument Making], Moscow, Mashinostroyeniye, 1989, 301 pages.

4. "Gibkiye proizvodstvennyye sistemy Yaponii" [Flexible Manufacturing Systems of Japan], edited by L.Yu. Leshchinskiy, Moscow, Mashinostroyeniye, 1987, 229 pages.

5. N. Komoda, K. Kera, and T. Kubo, "An Autonomous Decentralized Control System for Factory Automation," COMPUT., Vol 17, No 12, 1984, pp 73-83.

6. M.P. Galperin, Yu.A. Maslennikov, S.A. Nesterenko, and A.E. Reznik, "Small Local Networks of Microcomputers for Flexible Computerized Manufacturing Systems," AVTOMATIKA I VYCHISLITELNAYA TEKHNIKA, No 5, 1984, pp 42-45.

7. G.G. Bondarovich, A.V. Olekhnovich, V.V. Plotnikov, and D.A. Rossoshinskiy, "A Process Control Operating System for the Elektronika-60 Microcomputer," ELEKTRONNAYA TEKHNIKA. SERIYA EKONOMIKA I SISTEMY UPRAVLENIYA, No 4, 1982, pp 20-23.

8. G. Peterson, "The Theory of Petri Nets and the Simulation of Systems," Moscow, Mir, 1984, 264 pages.

9. V.G. Tsyvinskiy, V.V. Plotnikov, D.A. Rossoshinskiy, and A.V. Olekhnovich, "The Software of a Control Computer Complex Based on the Elektronika-60 Microcomputer," ELEKTRONNAYA TEKHNIKA. SERIYA EKONOMIKA I SISTEMY UPRAVLENIYA, No 4, 1982, pp 24-27.

10. D.A. Rossoshinskiy and L.A. Kovalchuk-Khimyuk, "Two Methods of the Exchange of Messages Among Processes in the Operating System for Microcomputers," UPRAVLYAYUSHCHIYE SISTEMY I MASHINY, No 2, 1987, pp 83-87.

Software in the Area of Evaluating, Monitoring the Reliability Indicators of Industrial Production

907G0130A Moscow NADEZHNOST I KONTROL KACHESTVA in Russian No 2 Feb 90 pp 60-61

[Text] In 1989 development was completed on a program complex to evaluate and monitor the reliability indicators of the products of machine and instrument building using the results of testing (NADIS version 1.0). The program complex is primarily intended to carry out the following tasks: choosing the nomenclature of reliability indicators; planning determinative testing for reliability; evaluation of the reliability indicators according to the results of binomial studies of the product as a whole or of its components; according to testing plans with a change in the operating time of the product as a whole or its components; according to statistical models of the "load-durability" type; according to dynamic models of the plans with a change in the determinative parameter; and verification of the reliability indicators according to the plans of one-stage and sequential monitoring at two monitoring levels, and according to the plans of one-stage monitoring at one monitoring level.

The guaranteed period of use of the NADIS complex is one year, during which the developers will correct errors detected in the program product.

On user request the NADIS program complex may be equipped with supplemental programs which expand its capabilities.

The program complex was implemented for personal computers compatible with the IBM PC XT/AT. It is the most complete program product developed in the Soviet Union in the area of the evaluation and monitoring of reliability. It can be used by reliability and quality control services to solve the problems of assuring reliability.

The main sources for the creation of the program product are the materials of reference works on reliability edited by I. A. Ushakov, O. I. Teskin, and R. S. Sudakov, as well as normative technical documents on reliability.

The program modules are united by a controlling program which provides maintenance and interactive services. The presence of a well-developed system of prompts makes it possible for nonprofessionals and beginning reliability personnel to use this complex. The teaching software can also be used in institutes to increase skills, and in institutions of higher learning.

Version 1.0 of the NADIS program complex will be distributed in January 1990 by the joint Soviet-British-American enterprise Spektrum (Ul. K. Marksa 19, 107066 Moscow).

Using Personal Computer in Elaboration of and During Command and Staff Exercise

907G0066 Moscow VOYENNO-MEDITSINSKIY ZHURNAL in Russian No 7, Jul 89 pp 14-15

[Article by Candidate of Medical Sciences, Medical Service Colonel E.P. Petrenko, Medical Service Lieutenant Colonel Yu.Ye. Gavrilenko and Medical Service Captain Yu.L. Minayev]

[Text] It is well known that in the process of decision making and planning of medical support services for an upcoming combat (operation) a combined unit's (formation's) medical officer performs a number of computations (the number of battle casualties and the demand for medical service personnel and resources) and uses both absolute and relative (intensive and extensive) indices. The majority of the latter have clear algorithms, their computation is routine in character and takes a lot of time if automated means are not used. The computations do yield themselves to computer processing, which makes it possible for the medical officer to concentrate on creative aspects of his or her work which considerably speeds up the process of evaluation of situations and decision making.

In the 1987-1988 academic year, an attempt was made at the Military Medical School of Kuybyshev Medical Institute imeni D.I. Ulyanov to use personal computer "Elektronika BK-0010-01" and programmable microcalculator "Elektronika B3-34" in the elaboration of and then during a command and staff exercise with the faculty. The computer is a microcomputer. It has the speed of 300,000 operations per second and a 32 KB ROM and uses a regular TV monitor as the display. Programs for the microcomputer are recorded on microcartridges, information from which is entered into computer's ROM with the help of a tape recorder. Communication with the computer is conducted in a dialog mode, which provides for data entry from the keyboard by answering questions displayed on the screen.

Software was the main problem. In order to compile the program, the exercise developers prepared a system of the logical sequence of medical officer's actions with specific computation algorithms. The program was written in Basic; it is approximately 200 lines long, which had taken the programmer around two weeks. The program made it possible to calculate the number of casualties, taking into account the specific theater of operations, various directions of enemy's main attacks and distribution of battle casualties among combined and individual units; it also makes it possible to compute the required medical service personnel and resources based on the number of battle casualties in a specific area. Using the program, the developers were able to perform all necessary calculations and print out medical support services charts in a matter of minutes.

The capability to "look up" alternatives of formation of battle casualties, simulate various situations of organization of medical support services for the battle (operation) and choose the optimum organization for the exercise. The program, which was recorded on a compact cartridge, did not include any classified data, which is extremely convenient in practice.

During the command and staff exercise, the medical officer, principal medical specialists and other officials effectively used a BK-0010-01 personal computer equipped with the above program; they only specified the level of casualties and their extensive distribution among combined units at a given period in time. The computer displayed the answer on the screen, both in terms of the absolute number of types of casualties and in terms of the demand for transportation means of evacuation, physicians teams, medical equipment and supplies etc.

Programmable microcalculator B3-34 was also widely used during the exercise, in accordance with procedures developed at the Military Medical Academy imeni S.M. Kirov. Time savings realized by exercise participants made it possible for them to comprehensively think over and make the most expedient decisions.

In our opinion, the above working method, as well as the program that was developed during the preparation of the command and staff exercise, can be recommended for application in combat troops.

Basic Directions of Informatization in the UkSSR Minobrazovaniye System

907G0046A Kiev MEKHANIZATSIYA I AVTOMATIZATSIYA UPRAVLENIYA: NAUCHO-PROIZVODSTVENNYY SBORNIK in Russian No 4, Oct-Dec 89, pp 20-23 [MS Received 20 Mar 89]

[Article by V.Ye. Bykov, candidate of economic sciences, and A.P. Osadchuk]

[Text] The main goals of informatization in the UkSSR Minobrazovaniya system are:

To implement the comprehensive automation and computerization of the educational process, of scientific research work, and of sectorial management processes;

To organize the production and introduction of a software product in the educational institutions of the UkSSR Minobrazovaniya system;

To assist in introducing computer equipment in cultural and educational activity and in organizing the population's leisure.

The basic directions of informatization are:

In educational activity: to consistently convert to computer-oriented technology for instructing pupils (differentiated by age groups) via the extensive provision of educational institutions with computer hardware and communications systems, including local computer networks with automated work places (ARM) using educational-purpose personal computers (PEVM), to develop automated instruction systems, trainers, audio-visual interactive computer devices and pedagogical program packages and to introduce them into the educational process; to increase the completeness of information support for pupils based on the use of interactive information-reference and information-consulting systems, the organization of access to knowledge bases, and the use of expert systems;

In scientific activity: to create effective computer tools systems for scientific research, to automate the system of scientific pedagogical information, and to create knowledge bases and expert pedagogical forecasting systems;

In administration: to develop a standardized computer-oriented technology for collecting and processing sectorial planning and book-keeping information; to create and introduce a sectorial computer network, a distributed sectorial data bank, sectorial knowledge bases (of an administrative nature), and interactive means for access to them by administrative employees, and to organize interaction with intra-sectorial ASU; to create and introduce a system for the imitative modeling of sectorial management processes and expert systems for forecasting the consequences of administrative decisions, to introduce "electronic mail" and "business games," and to create automated work places for administrators; to automate the processes of designing automated systems;

In producing software for educational institutions: to organize a sectorial network of software centers, to gather and analyze, develop, put together and distribute pedagogical software; to automate software development processes;

In cultural and educational activity: to ensure the broad use of computers in extracurricular work with pupils, to create computer clubs and centers for scientific and technical creativity, to organize the availability of scientific and technical services for the application, mastery and utilization of computer hardware and of computer and business "games," as well as for universal computer education, through cooperatives or other cost-accounting organizations.

The informatization of educational activity is especially relevant.

Computer-oriented technology for instructing the most numerous category of pupils, those in general educational schools, calls for:

Familiarizing pupils with theoretical material on a school subject (physics, mathematics, biology, Russian language, Ukrainian language, basics of information science and computers, etc.);

Controlling the course of the pupils' process of learning the textbook theoretical material (pupils' self-control);

Individualizing and personalizing the process of learning the textbook theoretical material;

Individualizing and personalizing the automatic selection of a package of quiz assignments and exercises for reinforcing the theoretical materials being studied;

Controlling the course of performance of quiz assignments and exercises;

Managing the process of computerized instruction in a subject on the part of the teacher;

Evaluating the quality of knowledge, obtained in a lesson on the subject being studied, and preparing statistical information for the period of interest;

Testing pupils for purposes of professional orientation;

Familiarization with additional information on the subject being studied (its section, theme), in information retrieval system (IPS) mode;

The possibility of encouraging pupils' participation in computer games for better assimilation of lesson material;

The creation of a set of exam materials and preparation of exam questions on a subject, using a random number generator.

The specific functional features of the pedagogical software (PPS), being developed to accomplish the above purposes, are:

Its orientation toward operation with personal computers with highly developed software (the IBM PC XT, IBM PC AT computers or ones similar to them), as part of a local computer network;

Support of stable on-line communications between pupils and the computer, between pupils and the teacher (inquiries in the order received), between the teacher and each pupil (by showing background information on the pupil's display screen or by transmitting the content of the pupil's screen to the teacher's display);

The possibility of operating in the "instruction," "quiz assignments," "exam," "statistics," or IPS modes.

The operating conditions for the PPS stipulate:

The provision of an opportunity for the teacher, when necessary, to interrupt the course of the educational process at a pupil's work place;

The availability of the necessary documents, teaching and instructional materials;

The possibility of displaying information on a large screen which all students can view;

Maintaining anonymity when releasing statistical data;

The use of computer time should take into account the sequence and complexity of the school subjects being studied during a day.

The following technical requirements determine the quality of the PPS:

Ensuring the stable operation of personal computers at each of the 12-15 ARM for the pupils when studying school subjects using the self-control mode during instruction, and ensuring automatic maintenance of "pupil-teacher" interaction;

An orientation toward using educational databases: databases and the means of access to them should be separate from each other;

Ensuring the protection of programs, databases and working files from unauthorized access and improper circulation;

Inclusion in IPS of modules for controlling the content of databases and execution of programs, for predicting the possibility of appearance of errors and for exposing their causes;

Extensive use of existing basic software for the IBM PC XT and AT computers, particularly information retrieval systems, text and graphics editors, electronic tables, "prompting," "windows" and background information, a light pen, etc.;

Implementation of access to database information using a menu;

Accessibility of PPS for students, ease of reference to the PC and of interaction with it.

A system for computerized instruction of school subjects includes (according to the size of the project) 12-15 personal computers which are the pupil ARMs, one personal computer which is the teacher's ARM, and audio-visual devices and screens for information output.

An ARM includes a microprocessor-based systems unit, a keyboard, Winchester storage, a monitor, a dot matrix printer, and a collection of textual documentation.

The speed of the PC included in the ARM is 10-100 x 10^6 operations per second; the main memory size is 640-1000 kbytes, and the external memory size is 20 to 60 Mbytes.

When teaching theoretical material, four levels are used for the presentation of material: the first is intended for a pupil with poor progress; the second—average progress, the third—for pupils with good progress; the fourth—for pupils who are strong in the subject.

Theoretical material is reinforced by the solution of quizzes and exercises, which are selected in automatic mode depending on a pupil's level of progress and preparation (which corresponds to the level of presentation of theoretical material): at the first level—problems, the answers to which are determined by selection from among alternative answers; at the second—problems which require ascertaining the mistakes being made; at the third—problems which require designing algorithms and programs and establishing cause-effect relations; and at the fourth—tasks requiring imitative modeling and decision-making.

The correlation between theoretical material and the quizzes or exercises is 1:1. When presenting theoretical material, the core is singled out—the basic concepts, rules, abilities and skills which the student should master during the process of computerized instruction. Comparison of the pupils' answers to a sample is done based on criteria, not a norm, and evaluation of knowledge is done using a point system, rather than a percentage scale.

The use of PPS contributes to the pupils' development of an ability to formulate the process of studying material as an algorithm with a clear establishment of limitations (statements), source data and results. The time for an answer during "student-computer" interaction should not exceed 2 seconds; the duration of a lesson when using PPS should be 45 minutes, and short breaks (2.5-3 minutes) should be held after 15 minutes of lessons in order to rest the eyes.

The methodological requirements of a PPS are:

An orientation toward the sequential computerization of educational courses and, in the future, of the whole educational process. The teacher plays the role of educator and mentor;

Ensuring the display output of high-quality, content-rich and graphic information for analytical work by the pupils;

The use of high-level programming languages (BASIC, PASCAL, C and others).

Moreover, the PPS should make it possible to encourage good work by the pupils and to help them shape confidence in their knowledge and abilities.

With the development of a technical base and communications channels, school computer networks will gradually be connected to knowledge (data) banks, supported by a high-performance computer.

Development by the UkSSR Minobrazovaniya GVTs of the I88-111 package of pedagogical programs, intended for use with the "Yamaha" computer (Japan) and currently being used at a number of general educational schools, is the next step on the path to creating modern software for the computerized instruction of school subjects.

The informatization of administration remains important. Putting a second line of sectorial ASU (OASU) into use has made it possible to automate most administrative functions related to planning and the calculation of labor and material resources at a high level of sector administration. The basic trend in the development of OASU is to integrate all its levels and links.

The development and introduction of new information technologies, the automation of scientific pedagogical research, and raising the effectiveness of administrative decision-making will contribute to the further development of national education.

Automated Information System for Ensuring Operational Reliability of Dust and Gas Collecting Installations

907G0068 Moscow NADEZHNOST I KONTROL KACHESTVO in Russian No 11, Nov 89 pp 56-59

[Article by A.A. Dotsenko, B.N. Skobelskiy, I.S. Vinogradova, Ye.F. Amanatidi]

[Text] Due to known consumption tendencies in the economy, little attention has been devoted to raising the reliability and quality of dust and gas collecting installations, which provide for protection of the atmospheric environment from harmful industrial emissions.

One of the most promising and effective methods for ensuring reliability relates to the creation of integrated automated information systems for the gathering and processing of operational information on reliability. At the present time, such systems are well-known in the aviation and tractor industries, in nuclear power engineering, and others [1].

In order to protect the air from harmful industrial emissions, it is necessary to create a system, which provides for intra-sector and intra-state exchange of information on the reliability and quality both of dust and gas collecting equipment, as well as of installations which use them and are being built. The basis for the concept of the proposed system is a concept of it as feedback in a quality control system for the devices and installations of dust and gas collection of industrial [wastes].

For this purpose, an automated expert system should be created, based on a knowledge base on the reliability of quality of dust and gas collecting objects. However, it is more realistic to speak of creating several information systems. In the first place, it is necessary to create a system for the collection and processing of operational information about the reliability of dust and gas collecting equipment, consisting of the following components: a base and a system for management of a database on operational reliability of dust and gas collection, the applied program package "Automated System for Research Reliability," [guiding, managing] documents of the system for collection and processing of information on the reliability of dust and gas collection, a YeS computer with a main memory volume of no less than one Mbyte. The system should be oriented toward solving problems for the USSR construction materials industry, but the field for its application may turn out to be far broader.

At the present time, a network model of data on operational reliability of dust and gas collection has been developed, on the basis of which a database was created using the KARS database management system (DBMS). Data processing methods and algorithms have been developed for obtaining quantitative estimates of reliability indicators and of qualitative characteristics of dust and gas collecting equipment and its components, as well as the first [version, edition] of the [managing, guiding] documents for the system of gathering and processing information on reliability, and experimental gathering and processing of data is being carried out. In particular, dust and gas collection reliability has been analyzed for installations based on electrofilters. A specific feature of the chosen methodological approach is the use of the parameter for flow of failures as the observed indicator, which makes it possible to reduce manual operations to a minimum in the gathering and statistical processing of data.

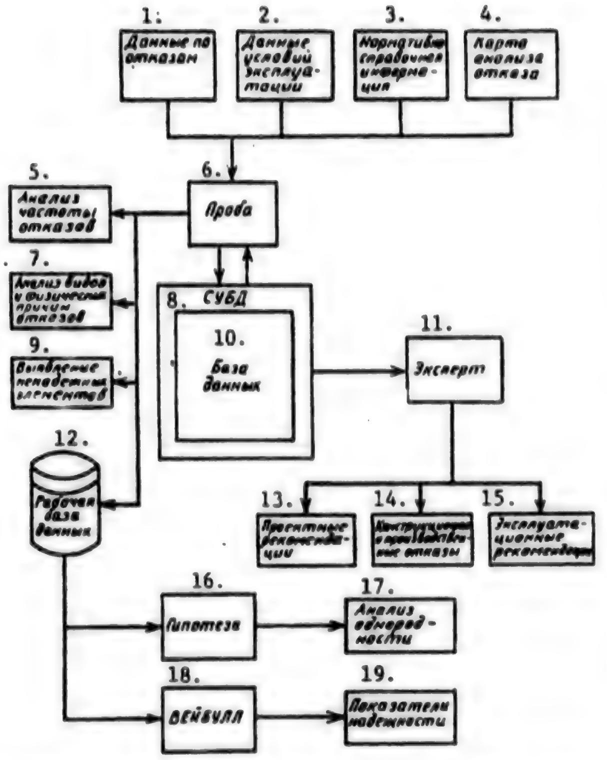
Requirements made of the functional characteristics of such a class of systems are based on a study of the components of the process of implementing specific data processing tasks [2]. The following components of the process can be singled out: formation of an information base for the task (system); technological processing of files and data; [content] processing of data; and conclusion of results of [solving the task].

Let us examine in more detail the structure of an automated system for reliability research, given in the diagram and intended for pre-design analysis of the reliability of equipment used when building dust and gas collection systems. The system is implemented on the basis of the KARS DBMS and consists of four sets of programs: "Proba" [test, sample]—input and preliminary processing of data; "Gipoteza" [hypothesis]—testing of hypotheses on the homogeneity of selective aggregates; "Veybull"—determination of the characteristic of failure-free performance of components for calculation of reliability; "Ekspert"—an expert analysis of equipment failures.

The software of the "Proba" complex supports, jointly with the standard [tools] of the KARS DBMS, the input and monitoring of input documents (data on failures of dust and gas collecting installations, conditions for use and normative reference information), management of databases and preliminary processing accumulated data with output of results in the following forms: analysis of frequency of failures, analysis of types and physical causes of failures, and exposure of unreliable components.

Data on failures of dust and gas collecting installations contain a code for the plant, number and degree of installation, data failure, description of malfunctions, types and causes of failures, and methods, time and labor outlays required to restore operability of installation.

In data on conditions for operation, the temperature of outgoing gasses, hydraulic resistance, quantity of gases in input and output, dust content, and equipment utilization coefficient are indicated.


1: Данные по отказам
2. Данные условий эксплуатации
3. Нормативно справочная информация
4. Карта анализа отказа
5. Анализ частоты отказов
6. Проба
7. Анализ видов и физических причин отказов
8. СУБД
9. Выявление ненадежных элементов
10. База данных
11. Эксперт
12. Рабочая база данных
13. Проектные рекомендации
14. Конструкционные и производственные отказы
15. Эксплуатационные рекомендации
16. Гипотеза
17. Анализ однородности
18. ВЕЙБУЛЛ
19. Показатели надежности

Key:—1.Failure Data—2.Data on Operating Conditions—3.Standard and Reference Information—4.Failure Analysis Chart—5.Analysis of Failure Frequency—6."Proba"—7.Analysis of Types and Physical Causes of Failures—8.DBMS—9.Finding Unreliable Components—10.Database—11."Ekspert"—12.Working Database—13.Design Recommendations—14.Structural and Production Failures—15.Operating Recommendations—16."Gipoteza"[Hypothesis]—17.Analysis of Homogeneity—18.VEYBULL—19.Reliability Indicators

The normative and reference information consists of 15 reference manuals, containing the names and classifications of objects.

Preliminary data processing is implemented using an order, in which the values of [query control characters] for selection of data from the database are indicated.

The "Gipoteza" program is intended for [paired] testing of homogeneity of two [selections, samples] according to an x-square criterion. The source data for operation of the program are the results of the processing of the two [samples] by the "Proba" complex of programs.

The "Veybull" program is intended for determining the parameters for [Veybull] distribution for the time between equipment failures according to an empirical dependency of the parameter of the failure flow on the time period for operation, obtained by the "Proba" program.

The "Ekspert" program is intended for processing maps/charts of an expert analysis of failures, containing the type of equipment, name of manufacturer, code for source of information, data failure, outward manifestation of failure, type of diagnostics, time and labor intensity for finding malfunctions, codes for parts and their elements, types, reasons, and frequency of failures, and methods, time and labor outlays for restoration of operability, technical nature of failures, qualitative requirements for ensuring reliability, and recommendations on operation and repair located in the database. Requirements for ensuring the reliability during design of dust and gas collecting systems, typical failures, methods for eliminating them and for preventing them during operation are shaped as a result of the program's operation.

Experimental use of the system on a WeS-1046 computer has shown good temporary results and the flexibility of information support.

Bibliography

1. Kubarev, A.I., Aronov, I.Z.; Burdasov, Ye.I., [Standardization of Methods for Gathering and Processing Information on Reliability]. Izdatelstvo VNIIKI, Moscow, 1985, 60 pp.

2. Cherkasov, Yu.M., [Automation of ASU Design Using Applied Program Packages. Energoatomizdat, Moscow, 1987.

UDC 656.7.07:658.386:159.955

Expert Systems in the Training Process

907G0152 Moscow VOZDUSHNYY TRANSPORT in Russian, Dec 1989 (manuscript received 14 Dec 1989) pp 1-31

[Article by G.Yu. Mitrofanov]

[Abstract] This is an overview of the role which expert systems and expert-teaching systems can play in the training process. Recommendations are made on the features expert systems should have to facilitate learning. For example, it should be simple and convenient to use, be able to explain its actions, and model interaction with a student for later diagnosis. It should offer instruction in natural language using the terms of the subject field. A proposed structure for such a system is discussed which includes several knowledge bases and utilities for the student, such as a help system, a system to explain errors, and a system which tracks the progress of individual students.

Desirable hardware features (universality, power, efficiency, ability to perform specialized tasks) are discussed briefly. A large portion of the work is devoted to the description of Western programs and languages used to generate, organize, and modify expert systems. Languages discussed include AIMDS, LISP, PROLOG, SMALLTALK, and others. Another section is devoted to

Western expert system packages in specific fields, including medicine, business, chemistry, computing, microelectronics, etc.

The conclusion describes the recent boom in intelligent systems and problems to be solved (i.e., software to be developed) before industrial intelligent systems can be introduced. Figures 1; references 24: 16 Russian 8 Western.

UDC 681.3.06+512.644+519.852

Package of Application Programs to Study and Solve Linear Problems

907G0097A Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1 Jan 90 pp 102-106 [Manuscript received 25 Jan 89]

[Article by V. N. Dzekh, L. M. Zaytseva, V. Yu. Kudrinskiy, V. Ye. Truten. Copyright: Izdatelstvo "Naukova Dumka" "Upravlyayushchiye Sistemy i Mashiny", 1990]

[Text] A package of application programs to study and solve systems of linear algebraic equations and linear programming problems (SLAU-2 ATL3.060.330) was developed at the "Voskhod" scientific research institute in conjunction with the Ukrainian SSR Academy of Sciences V. M. Glushkov Institute of Computer Science. The package is based on the well-known DISPLAN system [1], the SLAU [systems of linear algebraic equations) package [2], and basic system software. The package is intended to solve a wide range of scientific, technical, and economic problems which are formalized as problems of studying and solving various classes of systems of linear algebraic equations, systems of linear inequalities, problems of linear programming, linear multi-criteria programming, distribution problems, and estimation of ranges and matrix determinants. Study of a problem means determining its correctness in the classical sense, that is, the existence, uniqueness, and stability of a solution to the problem to infinitely small perturbations of the initial data.

In practice, the question of stability (in solving problems on a computer) is very timely, and is associated with the resolution of the question of dependence of the problem, the stability to small (finite) perturbations of the initial data and to errors in rounding the results of intermediate calculations.

Many criteria are known for the characteristic of dependence of systems of linear algebraic equations and problems of linear programming [3-5]. However, the quantitative characteristic of dependence in practical problems cannot be seen as isolated from the accuracy of the assignment of initial data, from the method or algorithm used to solve the problem, or from the arithmetic features of the computer. Consequently, in practice, the only reliable criterion of dependence is sufficiently effectively accurate a posteriori estimates of the total error of the solution of the problem. The authors have obtained sufficiently effective a posteriori estimates of the solution for systems [6] and problems of linear programming [7] which were implemented in the SLAU and SLAU-2 packages.

The experience of using these packages confirms the effectiveness and reliability of the use of the obtained estimates of total error of the solution for study of the problems.

Problems can be solved with a given accuracy using the package if the initial data and the size of the computer allow it, or the appropriate diagnostic messages can be issued in the converse case.

In the package, problems are divided into well and poorly dependent problems, large and small problems, and problems with dense or disperse matrices.

Poorly dependent problems are those which do not have stable solutions to small (finite) perturbations of the initial data and to rounding errors in the results of intermediate calculations. Approximate solutions of these problems obtained from the computer do not have the required (acceptable) accuracy and often lack a physical sense, which makes these problems virtually unsolvable.

In the package a number of original algorithms are implemented which solve with acceptable accuracy a sufficiently wide class of poorly dependent problems.

Large problems are those which require the use of external memory in the process of solving them. In the converse case the problem is considered small.

Efficiency in the speed of solving large problems is achieved by maximal combination of the work of the central processor and the input-output devices while the problem is being solved [8]. The dimension of the problems to be solved are limited only by the capacity of the magnetic disks used. The identification of large and small problems is done automatically depending on the available operating memory.

Problems with dense matrices are problems with matrices that contain less than 40 percent null elements. If the percentage is greater, then the problem is one with disperse matrices. The efficiency of solving problems with disperse matrices is achieved not only by appropriate information coding, but also by using original algorithms which are fast and which economize on operating memory in the solution process.

Problems involving the solution of systems of linear algebraic equations are divided into the solution of one-value and multi- value solvable and unsolvable systems of linear algebraic equations.

The solution of multi-value solvable systems of linear algebraic equations means a normal solution, that is, a unique solution, the one which deviates least from the given vector in a Euclidean metric.

Solution of full range unsolvable systems of linear algebraic equations means solution in the sense of least squares (a linear least squares problem). Solution of unsolvable systems of linear algebraic equations with an incomplete range means a generalized normal (pseudo) solution, that is, the unique solution which implements the minimum square of discrepancy and the least deviation from the given vector in a Euclidean metric [9].

To solve unsolvable and multi-value solvable systems of linear algebraic equations, original algorithms are used [10] which are more effective in accuracy and speed than other known algorithms which reduce to a solution of systems of normal equations [4] or systems of singular expansion [11].

To solve poorly dependent single-value, multi-value, and unsolvable systems of linear algebraic equations, original algorithms are implemented which are based on the transformation of poorly dependent systems of linear algebraic equations into well dependent ones.

To solve systems of linear inequalities, an iteration algorithm of orthogonal projection is implemented [12].

To solve linear programming problems, an original algorithm of the method of possible directions is implemented [13] (an algorithm of sequential search for active limitations). The direction of movement toward the optimum occurs along the "edges" of maximum dimension, and along the gradients of the appropriate target functions by distribution of the matrices using the halo method.

To solve problems of linear programming with disperse matrices of limitations, an algorithm of sequential search for active limitations is implemented with multiplicative representation of the inverse matrix based on the halo method. This has a number of advantages over the classical variants of the simplex method, which are implemented in widespread packages of application programs of the MPS series: PMP-2 (MPS), linear programming in an automated control system (MPSX) MPR-2 (MPS-370), and MINOS. In each iteration it operates on the matrix of minimum dimension, which makes it possible to solve linear programming problems in a natural statement with any (acceptable or unacceptable) initial point, which makes it possible to increase the dimension and speed of the solution of problems.

To solve well dependent problems of linear programming of large dimension, an original iterative algorithm of orthogonal projection is implemented.

To solve poorly dependent problems of linear programming, a combined algorithm is implemented which is the combination of iterative and direct algorithms. First, the iterative algorithm finds a "good" approximation. Then, beginning with the obtained point, the sequential search algorithm of active limitations determines the optimal point.

Solution of multi-criteria problems is reduced to the solution of sequences of linear optimization problems. At first, the person receiving the solution gives his system of preferences, that is, he ranks the criteria and indicates possible ranges of change of each of the criteria, which are supplemented by a given set of alternatives. Then, the algorithm of sequential search of active limitations seeks the local optimum of the most important criterion in the set of alternatives which the requestor formed. The value of the most important criterion is fixed in the form of an additional limitation. Then a search is conducted for the local optimum of the second most important criterion, for the new acceptable set, etc. As a result we obtain a single point, the preferred Pareto-optimal solution. To obtain a more preferable solution, the person receiving the solution must change his system of preferences, that is, he must again rank the system of criteria or indicate new ranges of changes.

The described approach to solving problems of linear multi-criteria programming is natural, and the algorithm of sequential search for active limitations is a more effective means of implementing it than the classical algorithms of the simplex-method.

The efficiency in the solution of problems using the SLAU-2 package is obtained by a high degree of automation of the entire process of solving problems on the computer, beginning with statement of the problem and the construction of appropriate information models (preparation of initial data) and ending with mathematical analysis of the obtained results and subsequent documentation of the results.

The SLAU-2 package consists of six subsystems: a control subsystem, input language analyzer subsystem, dialog monitor subsystem, reference subsystem, data control subsystem, and application subsystem.

The control subsystem obtains instructions from the operating system using the EXEC command of job control language or from a user program written in Assembler, FORTRAN, or PL-1. It then organizes the work of all package subsystems.

The interface between the user and the package is done with an input language consisting of six commands: SLAU-2 assignment, end of step, end of assignment, problem, data, reference.

The last three commands include a wide set of parameters which make it possible for the user to send the package knowledge about the problem and data, and to refine a reference about the package (designation, commands of the input language, structure of the input data, etc.) or the user may obtain information about the problem to be solved.

The input language analyzer transforms the assignment in input language into a more internal representation, with syntactic and semantic monitoring of the process.

The dialog monitor makes it possible, in interactive mode using the reference subsystem, to learn about work with the package, to formulate assignments at the start-up for a specific problem and to carry it out.

The reference subsystem gives the user access to information and reference services.

The data control subsystem is a complex of language and program tools which make it possible to prepare initial data and to control initial, intermediate, and output data.

The initial data may be given in the external or internal format of the package or in the standard MPS-format for commercial American packages [14].

The more internal representation is comprised of sets of data with direct organization.

The external format is a record (coding) of initial data in the form of a vector along the lines (80 symbols each) of a continuous file. The elements of the file are separated from each other by a comma. The order in which the elements of the vector follow in the file may be disrupted (null elements are not coded, and initial data may be prepared in portions). In this case before the numerical value of the vector element one gives in parentheses its name or order number in the vector. To write several identical sequential numbers one can use the multiplication sign. If the vector belongs to a matrix, then its name or order number in the matrix is indicated.

The package makes it possible to immediately write all initial data in the internal representation, input it in parts, and then prepare it in internal representation, and correct data which has already been input.

To indicate to the package the required function to use with the data, the language of indicator maps is used, with which one can indicate one of the following functions: WRITE, INPUT, PREPARE, CORRECT.

One can also use the package to transform initial data into the form required for a specific algorithm (program): transposition, division into cells, condensation (packing) of disperse matrices etc., as well as issuing the results of the solution of the problems in a form which is convenient to the user (the output MPS-file, names and values of all and selected output data and absolute and relative error).

It should be noted that the initial and intermediate data, as well as the results, are written in direct access volumes in sets of data which are created automatically. Messages are issued on the created sets of data and their names. This frees the user from labor-intensive work in calculating the memory of each data set, and this expands the range of users and eliminates additional errors.

The application subsystem performs the following functions. It identifies the problem, that is, it considers the parameters (the number of lines and columns of the matrix, the percentage to which the matrix is filled with non-null elements, the number of criteria, etc.) and properties (dependence, confluence, disperseness, balance, etc.; these properties may be given by the user or determined by the package) of the problem, and determines the algorithm (program) to solve the problem, if the user does not explicitly indicate it. It plans the data, that is, it defines the problem and the required form of the matrix for the program and makes the appropriate data transformation. It determines the optimal size of physical records of data sets according to the available free volume of operating memory. It initiates the work of the application program.

All programs of the application subsystem are linked through a knowledge base on the specific problem and the computer in which it being solved.

The output data is written to the direct access volume and issued to an alphanumeric printer in a form suitable to the user. The solution of linear programming problems is issued in output MPS format.

The SLAU-2 package works under OS YeS version 6.1 and higher. The minimum operating memory is 250 kilobytes.

The program modules of the product are implemented in Assembler and PL-1. The package may work autonomously, or may be called from any user program written in Assembler, PL-1 or FORTRAN. The program modules are equipped with well-developed diagnostic tools.

The modular structure and program implementation of the package makes it possible to actively solve the problems of expanding and modifying the package with new problem and program modules.

The SLAU-2 package is a complete original domestic experimental development which does not have direct prototypes.

In terms of construction, the package is formulated as a software product, which allows industrial replication with delivery to the user of the following: magnetic tapes with the six libraries of the package, which contain the initial modules, the load modules, assignments to obtain the load modules from the initial modules, and an assignment to monitor that the package is operating correctly, certificate of specifications on the package subsystems, reference information, listings (printouts) of the work of all start-up assignments available in the package; and a set of use documentation in accordance with the requirements of the Unified System of Design Documentation.

The SLAU-2 package has a number of advantages over commercial American packages: a wider range of solvable problems, detailed classification of problems from the point of view of effectiveness in accuracy and speed of solution methods (the possibility of study and solution of poorly dependent problems which are frequently encountered in practice, as well as the operative study of the area of limitations on degeneracy), the presence of effective programs to study and solve large problems, and the solution of problems with any given accuracy; new means of studying and monitoring the accuracy of calculations, which overlap the capabilities of post-optimization procedures of commercial packages; and broad and convenient capabilities (including MPS format) for preparing and correcting data. The format of American packages is inconvenient, but it is included in SLAU-2 for commercial considerations. The problem-oriented distribution language is simple, and it is possible to access the package and its subroutines from a user program written in Assembler, PL-1, or FORTRAN.

A great contribution to the development and program implementation of the SLAU-2 package was made by T. N. Vasilenko, Ye. P. Yeremina, V. N. Kots, N. S. Podgoretskaya, A. P. Mitropan, R. V. Mashkina, L. S. Sheremetyeva, and G. V. Artyushenko.

BIBLIOGRAPHY

1. Glushkov, V. M. On Sequential Optimization in Linear Macroeconomic Models. USiM 1973 No 4 pp 1-7.

2. Package of Application Programs to Solve Systems of Linear Algebraic Equations. V. M. Glushkov, V. I. Drakin, B. S. Berezkin et al. USiM 1982 No 5 pp 104-106.

3. Faddeyev, D. K., Faddeyeva, V. N. "Vychislitelnyye metody lineynoy algebry" [Computational Methods of Linear Algebra] Moscow: Fizmatgiz 1963 460 pp.

4. Uilkinson, Dzh. "Algebraicheskaya problema sobstvennykh znacheniy" [Algebraic Problem of Eigenvalues] Moscow: Nauka 1970 425 pp.

5. Forsayt, Dzh., Moller, K. "Chislennoye rasheniye sistem lineynykh algebraicheskikh uravneniy" [Numerical Solution of Systems of Linear Algebraic Equations] Moscow: Mir 1969 125 pp.

6. Kudrinskiy, V. Yu., Truten, V. Ye. Agreement of Errors in the Solution of Systems of Linear Algebraic Equations on a Computer. ZHURN. VYCHISL. MATEMATIKI I MAT. FIZIKI 1982 No 1 pp 223-227.

7. Kudrinskiy, V. Yu., Truten, V. Ye. Estimates of the Accuracy of the Solution of a Problem of Linear Programming, in "Vychislitelnaya matematika v sovremennom nauchno-tekhnicheskom progresse" [Computing Mathematics in Current Scientific and Technical Progress] Kiev: IK AN USSR 1974 pp 333-339.

8. Dzekh, V. N., Kudrinskiy, V. Yu., Truten, V. Ye. Solution of Large Systems of Linear Algebraic Equations on the YeS Computer. KIBERNETIKA 1981 No 5 pp 75-77.

9. Morozov, V. A. "Regulyarnyye metody resheniya nekorrektno postavlennykh zadach" [Regular Methods of Solving Incorrectly Stated Problems] Moscow: Nauka 1987 239 pp.

10. Lyashko, I. I., Kudrinskiy, V. Yu., Ostapchuk, V. S. Method of Determining the Generalized Normal Solution of a System of Linear Algebraic Equations. DOKL. AN USSR Series A 1986 No 7 pp 16-20.

11. Voyevodin, V. V. "Vychislitelnyye osnovy lineynoy algebry" [Computing Foundations of Linear Algebra] Moscow: Nauka 1977 304 pp.

12. Yeremin, I. I., Mazurov, Vl. D., Astafyev, N. N. "Nesobstvennyye zadachi lineynogo i vypuklogo programmirovaniya" [Improper Problems of Linear and Convex Programming] Moscow: Nauka 1983 336 pp.

13. Zoytendeyk, G. "Metody vozmozhnykh napravleniy" [Methods of Possible Directions] Moscow: Izd-vo Inostr. lit. 1963 175 pp.

14. Murtaf, B. "Sovremennoye lineynoye programmirovaniye" [Modern Linear Programming] Moscow: Mir 1984 295 pp.

UDC 681.324+510.8+378.6

Automatic Learning System for Decision Making in Trade Planning

907G0097C Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1 Jan 90 pp 112-115 [Manuscript Received 6 Jun 88]

[Article by V. I. Bromov, V. M. Kuchmin. Copyright Izdatelstvo "Naukova Dumka" "Upravlyayushchiye Sistemy i Mashiny", 1990]

[Text] **Purpose and features**

The programming and technical capabilities of the computer make it possible to construct automatic learning systems which model the decision making process in the control of an economic process. The decision making process is conducted in interactive mode with an expert economist. The Gomelskiy Cooperative Institute has developed such a system on an SM-4 (SM-1420) computer with a basic OS RV version 3.0 operating system. The system teaches the planning of retail trade circulation, expenses of circulation, and other indicators of the activity of cooperative trade.

According to the classification outlined in [1], the automatic learning system for decision making in the planning the trade of consumer cooperation may be considered an expert learning system. The constructive features of the product include the following. The process of learning decision making is done in an information-advisory dialog with the computer; decision making is supported by economic-mathematical models and methods implemented in application program packages. The information resources of the automatic learning system are managed by the "Archive File Management System" (AFMS) database management system [2]. The principles of individual learning are implemented. Learning is based on real economic information.

In contrast to automated learning courses which are constructed only with problem-oriented programming languages (YaOK, AOS-VUZ, etc.) this system provides a flexible scenario of man-machine interaction. This type of scenario is implemented by two decision making centers (man and computer) in the learning system, and a set of alternatives in decision making and variants proposed for the analysis of the initial data.

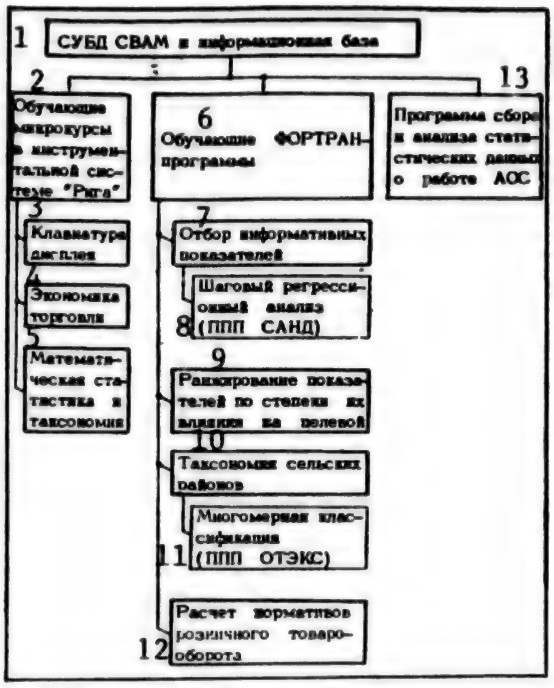
Structure and software

The programming medium of the learning system includes the AFMS database management system, an instrumental complex to construct "Riga" learning courses [3], the "Statistical Data Analysis" (SDA) application program package [4], and the "Experimental Data Table Processing" application program package [5]. Application of these programming tools to the specific subject field is done in the framework of the following automated courses: learning the concepts of mathematical statistics and taxonomy; learning the methods to plan retail trade circulation; learning to make decisions on the selection of indicators which characterize the conditions for conducting trade in the rural regions of an oblast and estimating their significance; learning to make decisions on multi-dimensional classification of rural regions; learning to calculate normative totals for trade circulation.

Programs are implemented to collect and analyze statistical data on the work of the students. The structure of the software for the automated learning system is shown in the figure.

The "Riga" instrumental complex is a general-purpose multi-console interactive system to construct learning courses. The learning scenarios are developed by an instructor of the special discipline and formulated in the form of a text file. The learning functions (presenting knowledge and monitoring it, provision of auxiliary material, repetition, etc.) are implemented by a simple and convenient choice of commands and directives. It is easy to introduce changes in the learning course. It is possible to analyze answers semantically.

Moreover, the Riga complex was chosen to construct the automatic learning system because both it and the application program packages work in the OS RV version 3.0 SM-4 (SM-1420) environment. This allows interaction with the AFMS database management system and the FORTRAN learning programs. This makes it possible to combine work with the automated learning system and the execution of other tasks on the computer. The learning course constructed with the Riga complex tools plays the role of a control program module, from which one may call programs written in various programming languages.


1
СУБД СВАМ и информационная база
2
Обучающие микрокурсы в инструментальной системе "Рига"
3
Клавиатура дисплея
4
Экономика торговли
5
Математическая статистика и таксономия
6
Обучающие ФОРТРАН-программы
7
Отбор информативных показателей
8
Шаговый регрессионный анализ (ППП САНД)
9
Ранжирование показателей по степени их влияния на целевой
10
Таксономия сельских районов
11
Многомерная классификация (ППП ОТЭКС)
12
Расчет нормативов розничного товарооборота
13
Программа сбора и анализа статистических данных о работе АОС

—1. AFMS Database Management System and Information Base—2. learning micro-courses in the "Riga" instrumental system—3. display keyboard—4. economics of trade—5. mathematical statistics and taxonomy—6. FORTRAN learning programs—7. selection of informative indicators—8. step by step regression analysis (SDA application program package)—9. ranking of indicators by their degree of effect on the target indicator—10. taxonomy of rural regions—11. multidimensional classification (Experimental Data Table Processing application program package)—12. calculation of norms for retail trade circulation—13. program to collect and analyze statistical data on the work of the automatic learning system

The AFMS database management system provides for the input, correction, storage and search for information needed by the user. In addition to the programming complexes which carry out the functions listed above and implemented in various similar information systems, AFMS includes the means of managing local dictionaries (classification systems), storing forms of input-output, model input and correction of information, generation of reports for issuing information. AFMS has a set of language and programming tools which interface with the application programs in the following programming languages: COBOL, FORTRAN, PASCAL, BASIC, C, and macroassembler. This make it possible to connect programs which implement specific algorithms.

AFMS is oriented toward work with users who are not programming specialists. Work with the system is carried out in interactive mode with a maximum of prompt options for the user to choose. This DBMS may be used to construct distributed local databases in interactive systems for collective use. The AFMS architecture is conceptually close to the architecture of the UNIBAD DBMS and also provides the appropriate environment for the end user.

The SDA application program package is designed to solve scientific, technical, economic, and other problems which require the use of mathematical statistics. It consists of three basic interconnected parts: a library of modules in the form of FORTRAN programs, which do not contain input-output commands; a library of algorithms to solve individual problems in applied statistics in the form of FORTRAN programs which allow the presence of input-output commands; and the DIALOG head FORTRAN program. The package makes it possible to combine and correct program modules to synthesize algorithms to solve problems.

The automated learning system uses an algorithm to estimate the parameters of a multiple linear regression using the step method. At each step a study is made of variables included in the regression equation in the previous steps of the solution. The variable which may be the best individual variable, which is worthy of introduction into the model at an early stage, may turn out to be unnecessary at a later stage due to an interrelation with other variables in the model. To verify this, at each stage of the solution, calculations are made of partial *F*-criteria for each variable of the regression equation, and they are then compared with an accuracy percentage of the *F*distribution which was chosen earlier. Any variable making an insignificant contribution is excluded from the model.

The "Experimental Data Table Processing" application program package is designed to solve problems which require the use of taxonomy methods and pattern recognition. The algorithms in this package are designed to work with "object-indicator" tables, which may have a random correlation between the number of objects and the number of indicators, indicators of various scales (nominal, order, arithmetic), or some absent values. A FOREL type heuristic taxonomy algorithm is used in the automatic learning system.

Functional capabilities. The automatic learning system models the structure of the decision making process for determining numerical values of normative retail trade circulation for rural regions of an oblast. The norms obtained may be used to validate plan assignments, for objective evaluation of the work of each collective and for material stimulation. The volume of retail trade circulation is, to a great degree, determined by those conditions in which people work. The norm of retail trade circulation should be totally defined by objective factors, independent of the will of the people.

The regions of the oblast are grouped using taxonomy methods by objective conditions of management into nonintersecting classes. Each class is formed from rural regions with similar characteristics. The group average arithmetic sale of goods per capita is taken as a norm.

Indicators which define the conditions of management (socio-economic, transport-geographic, demographic indicators) are stored in a database under AFMS.

Basic micro-courses in economics and mathematical statistics precede the learning sections. They make it possible to remind students of the basic concepts of these disciplines and verify their knowledge. The micro-courses are written using the Riga instrumental system.

Learning to make decisions involves a set of alternative decisions and the choice of the best of these using a specific criterion. At the first stage, when the most informative subset is chosen from the set of various indicators, the computer then generates a decision making situation: the student evaluates the information content of the next indicator considering its effect on the target indicator and interrelations with other explanatory indicators. A preliminary diagnostic decision is made based on the personal experience and knowledge of the student. Then the student can request additional information on the indicator (the average arithmetic and standard deviation, coefficient of variation, paired correlation coefficients). Considering the new data on the indicator the student must confirm his previous decision.

The computer issues its own decision, which is obtained by step-by-step regression analysis (the SDA application program package). The final decision on the inclusion or exclusion of some indicator remains the task of the student. He may include the indicator with other informative indicators if he thinks that it improves the explanation of the oscillations of the target indicator, or, conversely, he can exclude the indicator if the link is, in his opinion, random or erroneous.

As a result of the first stage, a subset of only informative indicators is separated from the total set of indicators. In the second stage, the student evaluates the effect of each indicator on the value of the target indicator. In the course of a dialog with the computer, the student assigns the weight of the selected indicators, compares them with those obtained based on standardized regression coefficients, and makes a final decision.

The ranked weights of indicators are used to divide the rural regions into homogeneous groups. The classification of regions by management conditions is the final step of learning in the automatic learning system. As in the previous stages, the student encounters a multivariant choice. The student varies the number of classification groups, aiming for the best division. The computer issues a statistical evaluation of the quality of the division according to the Fisher criterion, which is calculated from the values of the target indicator.

Moreover, the student is guided by a wealth of considerations. Calculation of the desired norm is based on the use of group average actual values of the retail trade circulation by rural regions of the oblast; thus, the issue of group numbering is significant. The student considers that a large number of groups usually leads to a decrease in the statistical confidence of the results of norm calculations. The groups turn out to be small in number, and factors which are not related to objective conditions of management begin to have a significant effect on the retail trade circulation. A small number of groups leads to a leveling of individual differences. Even sharply different regions obtain the same norm, which places them in conditions which are known to be unequal.

The student has the opportunity to form various chains of decisions by changing the indicators used in classification, their weight, and the number of groups. Returning to the earlier stages of learning makes it possible to track how the results changed in relation to the decisions made, and to implement the "what if" principle.

Due to the discussion character of decision making and the use of real data, the student adds more depth to his knowledge and understanding of the mechanism of interrelation of socio-economic indicators which characterize the object of study. Activation of one's own knowledge in the gradual formation of a final decision, and the quantitative analysis of real data frequently helps change traditional concepts of economic rules, and breaks down developed stereotypical thoughts.

For active tracking of the course of learning and the collection of statistical data, a work log is formed and issued. In it, the decisions of the student and of the computer are reflected at each step of the work, and a total evaluation is given.

Computerization of the learning process in various interconnected disciplines requires the formation in institutions of higher learning of a single automated information resource. In our institute the information resource is formed from real socio-economic information. The reality of the data with which the student works makes it possible to develop his interest in the object under study and to bring the learning process closer to practical economic work, and, in the final analysis, accelerate the adaptation of young specialists after they leave institutions of higher learning.

Over the course of time the database will accumulate the values of socio-economic indicators over several years and various regions. These data are used for research and automated construction and accumulation of object models. The logic of the process of constructing models of objects is as follows.

From the entire set of indicators *P* which define the conditions of management, the expert economist chooses a subset of indicators *P1* (*P1* is a subset of *P*) which are characteristic for a given region. Then for a fixed *P1*, step-by-step regression analysis in a dialog with an expert economist is used to isolate a subset *P2* (*P2* is a subset of *P1*), whose indicators best describe the target indicator.

It is obvious that the content of subset *P1* for various regions of the country will coincide almost completely. The differences arise basically due to transportation and geographic criteria (the presence of sea transport, railroad connections, etc). At the same time the regression equations constructed in subsets *P2* may have substantial differences for different regions. The base of models in the form of linear regression equations is used by the students to learn about regional characteristics of the organization of cooperative trade.

The automatic learning system was tested in simultaneous work with ten standard terminals on an SM-1420 computer. The reaction time of the system for each terminal did not exceed 3 seconds. For a fixed set of data the number of alternative situations created in the course of learning was no less than $N!*M!*(K-2)$ where N is the total number of indicators, M is the number of indicators chosen to construct the regression model, and K is the number of regions of the oblast.

BIBLIOGRAPHY

1. Dovgyallo, A. M., Yushchenko, Ye. L. Learning Systems of the New Generation. YCiM 1988 No 1 pp 83-86.

2. "Sistema vedeniya lokalnykh arkhivnykh massivov: Opisaniye primeneniya" [Archive File Management System: Description of Use] Riga: NIIP Gosplan Latv. SSR 1987 95 pp.

3. "Metodicheskiye rekomendatsii po programmirovaniyu uchebnykh distsiplin avtomatizirovannoy obuchayushchey sisteme 'Riga'" [Method Recommendations for Programming of Educational Disciplines in the "Riga" Automated Learning System Riga: MIPKSNKh Latv. SSR 1984 68 pp.

4. "Paket prikladnykh programm 'Statisticheskiy analiz dannykh' (PPP SAND): Opisaniye primemeniya" ['Statistical Data Analysis' Package of Application Programs PAP SDA: Description of Use] Kalinin: NPO "Tsentrprogrammsistem" 1987 128 pp.

5. "Paket prikladnykh programm OTEKS (dlya analiza dannykh)" [The OTEKS Package of Application Programs (for Data Analysis)] N. G. Zagoruyko, V. N. Yelkina, S. V. Yemelyanov, G. S. Lbov. Moscow: Finansy i statistika 1986 160 pp.

Software for the New Information Technology

907G0133A Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 1, Jan 90 pp 124-125

[Article under the "Chronicle, Information, Letters to the Editor" rubric by N. A. Semenov]

[Text] From 24 to 26 October 1989, in Kalinin, on the grounds of the Tsentrprogrammsistem NPO [Scientific Production Association] of the USSR Ministry of Electrical Engineering Instruments, the first All-Union Scientific and Engineering Conference "Software for the New Information Technology" was held, organized jointly by the Science Council of the USSR Academy of Sciences on the subject of artificial intelligence and the Soviet Association for Artificial Intelligence. Taking part in the proceedings were more than 250 scientists and specialists from 102 organizations in 42 cities of the Soviet Union.

The goal of the conference was to analyze the current status and determine the future trends in development of software for the new information technology (NIT), to exchange results of theoretical and applied research in the field of NIT, and to discuss various aspects in the creation of artificial intelligence (AI).

Eight reports were given during the plenary sessions.

The conference was opened by Deputy General Director of the Tsentrprogrammsistem NPO, N. A. Semenov, who gave a report "Status and prospects for research of the Tsentrprogrammsistem NPO in the field of artificial intelligence." Since 1986, in keeping with the goals of the Comprehensive Program of Scientific Technical Progress (KP NTP) of the CMEA Countries for 1986-1990 and the Period up to the Year 2000, the Association has been working on the development package (DP) LOGOS in the environment of MS DOS of IBM type PCs, which is designed to create "empty" expert systems (ES). In the framework of the DP LOGOS project, the PUSHOK program package was developed and handed over to the GosFAP [State Holdings of Algorithms and Programs], being a tool for creation of learning programs. In keeping with the goals of the KP NTP, research is being carried out to develop LISP translators for minicomputer and PC. As regards the operational hardware of the YeS computers and PCs, the NPO Tsentrprogrammsistem has developed the program package (PP) EKSPERT, which belongs to the category of empty ES and is designed to create knowledge bases (KB) in the area of solving of diagnostic type problems. A wide array of research is being carried out at the Association in the context of maintenance of development tools for creation of ES, including the languages PROLOG, LISP, empty ES INTER-EKSPERT and the PP EKSPERT. There are plans to organize studies on the subjects of AI in the direction of developing application and hybrid ES in ASU [automated management systems], ASNI [automated scientific information management systems], and CAD systems, and also as applied to subject fields in the nonindustrial sector.

The report of the Chairman of the Science Council of the USSR Academy of Sciences on the Subject of Artificial Intelligence, Academician G. S. Pospelov (computer center, USSR Academy of Sciences, Moscow), "The new information technology," made a critical analysis of the state of research in the field of AI in our country and abroad. In particular, he argued for the need to automate the programming process in order to create intelligent information-retrieval and logical-computing systems, integrated or hybrid ES, in which the functions of logical inference and mathematical modeling are combined. The report examined problems associated with incorporation of the knowledge of experts and a procedure which may speed up this process by a factor of 8-10. He defined the concepts of a "hard" and a "soft" program and argued for the need to create soft programs, which are furnished with explanation and inference facilities, allowing the end user to watch the entire process of decision making and, if need be, to intervene.

The report of the President of the Soviet Association for Artificial Intelligence, professor D. A. Pospelov (computer center, USSR Academy of Sciences, Moscow), "Cognitive models in intelligent systems," was devoted to questions in the field of the structure of knowledge in the human brain and the change in the basic paradigms associated with knowledge. The left hemisphere of the brain performs symbolic information processing, the right hemisphere works with visual images, the frontal lobes control the subconscious and support the process of learning. Experimental studies have overturned the previous model which defined three kinds of memory: fast (0.5 s), short-term (7+/-2 s), and long-term (forever). The report examined the role of the scale of antonyms, the concept of the division of human consciousness into a number of noncontradictory worlds, introduced the concept of "soft" quantors, and performed an analysis on the types of representation of knowledge in the form of production systems, semantic networks, and frames. Unlike machine output, the human response to a question asked is not a retrieval, since a new decision synthesizing mechanism is involved each time. The report defined three paradigms associated with knowledge: formal systems and logical inference; open systems, in which inference is replaced by argumentation in terms of the available system of knowledge; and knowledge, tinged with certain judgments, and instead of inference or argumentation there is a justification of the knowledge in an existing system of values. Only the first two paradigms are being studied at present.

Various technologies for development of small, large, and industrial ES were defined in the report of L. I. Mikulich (Institute of Management Problems, USSR Academy of Sciences, Moscow), "A technology for design of expert systems." In the framework of the concept of the life cycle of an ES, the phases of their design, development, and maintenance were defined. The design technology, which we lack at present, is many times more valuable than the ES themselves. Unless the situation changes, the current vogue for expert systems will pass and they will become discredited in the eyes of the end users, similar to what happened with the ASU.

Professor V. V. Aleksandrov (LIIA [not further identified], USSR Academy of Sciences, Leningrad) gave a report "Methods of processing visual information in intelligent systems." The report made a comparative analysis of the present information science infrastructures in the USSR and the USA. At present, the relative number of professional programmers in our country is not more than 10% of the worldwide level. It is proposed to promote research in the field of creation of ES on an elite basis at major academic organizations and the leading technical colleges in the country.

The report of A. Ya. Dikovskiy (Institute of Mathematical Problems, USSR Academy of Sciences, Moscow), "Automation tools for the development of intelligent application interfaces," presented a classification of development tools for intelligent interfaces. It pointed out the special features of the interfaces of ES, which involve the explanations of the actual operations and answers to the questions of Why, How, What If, and so on. One of the contemporary methods of creating interfaces within an object-oriented approach was explained.

The report of A. A. Zenkin (Moscow State University), "Use of cognitive graphics in information systems," was devoted to an explanation of the concept of interactive computer graphics (ICG), which is able to show on the display screen, in vivid color and musical form, the essence of the most abstract scientific ideas, hypotheses, and theories. Deliberately acting by means of such ICG images on the right hemisphere (pictorial, intuitive thinking), it is possible to activate significantly the human creative capacities. A methodology has been developed for such ICG knowledge-generating technology, which has been used to create a man-machine ICG system, the Interactive System for Investigation of Abstract Problems in the Classical Theory of Numbers (DSTCh). New scientific findings have been reached by means of the DSTCh in the context of the classical Waring problem, which has been exhaustively studied in the [illegible] Euler, Lagrange, Gauss, Gilbert, Hardee, Vinogradov, and many other prominent scientists. The DSTCh system has been implemented on the TORCH PC and, in particular, the IBM PC.

The report of V. N. Agafonov (Tsentrprogrammsistem NPO, Kalinin), "Simple mechanisms in knowledge-based systems," advanced and supported the idea that simple knowledge presentation and decision making mechanisms are at work in the majority of knowledge-based systems in actual use (ES, learning systems, etc.). In the capacity of such mechanisms, the report considers decision tables, propagation ("electronic") tables, atomic formulas and simple implications, simple relational structures, networks of interdependencies, and certain kinds of equations. Attention is paid to the sufficiency of these mechanisms in many practically important situations and to the promise of the systems based on them.

Two round table discussions were organized at the conference.

Prof. D. A. Pospelov acted as discussion leader on the topic "The problem of development of a friendly interface in intelligent systems." Matters relating to implementation of systems for speech synthesis, visualization of situation patterns in decision making support systems, knowledge representation in computer vision systems, and implementation of an integrated interactive system of visual interaction with computer were discussed in the wake of a lively exchange of opinions.

A discussion on the topic "Knowledge-based systems" was organized by L. I. Mikulich. Matters relating to assessment of the quality of ES performance, the feasibility of making use of knowledge engineers (cognitologists), the creation of learning systems, several aspects pertaining to the differences and the common features of data bases and knowledge bases, and the need to develop hardware for knowledge-based systems were discussed.

The collection of papers of the conference published 139 synopses of the podium reports, discussion of which was organized in five sections: "NIT methodology: General topics"; "Methods of organization of a friendly interface"; "Data and knowledge bases"; "Software in information and management structures"; and "Expert and teaching systems in CAD and ASUTP [automated process control systems]."

During the proceedings at the Tsentrprogrammsistem NPO, the Czechoslovakian company Kantselyarske stroe [Office Machines] organized an exhibit and sale of software oriented to solving a broad range of engineering and administrative problems in various types of computers.

In the unanimous opinion of the participants, the conference was conducted on a high scientific and technical level, and it was a useful and fruitful exchange of research findings in the field of AI. It was resolved to hold the second such conference on the grounds of the Tsentrprogrammsistem NPO in October of 1991.

Synthesis of Structures of Automated Management in Large-Scale Systems

907G0134A Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 2, Feb 90, pp 123-124

[Article under the "Chronicle, Information, and Letters to the Editors" rubric by V. Yu. Khodakov and F. B. Rogalskiy]

[Text] From 12 to 14 September 1989, at Kherson, the All-Union Seminar on Problems of Synthesis of Structures of Automated Management in Large-Scale Systems was held, organized by the Science Council of the USSR Academy of Sciences on the Comprehensive Problem of

Computer Science, the Institute of Management Problems of the USSR Academy of Sciences, and the Kherson Industrial Institute.

At the seminar there was an exchange of opinions, research findings, and practical experiences on the following basic topics in this particular subject: optimization of the structural layout of automated information management systems, multiple-machine complexes, and communications networks; formalized description of elements of the systems and their structural interdependencies; decomposition of systems into subsystems, coordination and aggregation of system elements; simulation modeling and its use for analysis and synthesis of the structure of large-scale systems; and management of the development of the structures of large-scale systems.

Taking part in the proceedings were specialists representing various ministries, agencies, academic and trade scientific-research and scholarly institutes.

The detailed report "Problems of synthesis of structures of automated management in large-scale systems" was given by A. D. Tsvirkun (Moscow). The report clarified the stages of development of management systems and methods of synthesis of the structures of development of complex systems. It examined questions of mathematical modeling of the management of the development of structures of large-scale systems. Use of the methods of synthesis was illustrated by examples of problems of planning the development of organizational production-engineering systems for a number of industrial sectors.

V. F. Shostak (Kharkov) devoted his report "Support of decision making on the basis of simulation models and expert appraisals" to matters of improving the hardware and software for making managerial decisions and support thereof in one class of complex technological systems, characterized by a considerable number of parallel-sequential technological processes. A decision making procedure was proposed, based on integration of the elements of a knowledge base and simulation management models.

The report of B. A. Platonov (Kiev) "Use of the technological approach in synthesis of the architecture of distributed systems" examined several aspects of distributed information processing, based on the use of a technological approach and formalized utilities which are proposed for finding the solution to the problem of synthesis of the architecture of a prospective automated data processing system.

The report of V. Ye. Khodakov (Kherson) "Synthesis of structures of information service systems" proposed a new approach to the synthesis of the structures of information systems, known as the information-structural approach.

We should mention a number of the reports in the sections. B. L. Kuchin and Ye. V. Yakusheva (Moscow) formulated the priority issues of development of complex structures in the environment of scientific and technical progress, taking into account the dynamism of the latter. V. P. Kostyuk (Saratov) devoted his report to the use of coordination methods in the synthesis of the structure of multilevel hierarchical information management systems, giving special attention to practical implementation of this approach in the ASUTP [automated process control systems] of industrial enterprises.

The report of P. N. Pechatkin (Kherson) examined the creation of a flexible ASU [automated management system] for form generation processes, as well as the criteria for optimization of the intelligent interface of the system.

A large group of reports was devoted to a discussion of aspects of the synthesis of structures of automated large-scale regional systems of various configurations, automation of multivariant production systems, and development of program packages for automation of research and industrial operation (A. A. Sakhnin, V. K. Akinfiyev, Ye. N. Konovalov, Moscow; A. V. Karibskiy, T. V. Rylskaya, Moscow; G. P. Artemenko, Kharkov; V. I. Savelyev, A. V. Karibskiy, Moscow; V. Ye. Trush, F. B. Rogalskiy, Kherson; A. V. Gabalin, A. V. Karibskiy, Yu. R. Shishorin, Moscow; G. S. Yakimchuk, Kherson; S. N. Klimachev, Moscow; and others).

The attention of the participants was drawn by the reports of S. D. Bushuyev, V. V. Morozov, and D. Ye. Dekhteryuk (Kiev), which examined integrated systems of automated teaching of management of complex systems and aspects of their use in the educational process of the technical colleges, and that of F. B. Rogalskiy (Kherson) on the use of the information-structural approach to the synthesis of structures of management information-computing systems. In addition, a number of communications were heard on subjects at the borderline between various sciences.

The experience set forth in the above and other reports will doubtlessly be useful in elaborating the theory and practice of the synthesis of structures of automated management in large-scale systems.

The resolution of the seminar noted the following:

—the work devoted to methods of synthesis of structures of ASU in large-scale systems constitutes an independent scientific field with its own problems, methods, and object of investigation;

—use of the results obtained in this field to solve practical problems enables substantial improvement of the quality of the systems designed and in certain cases produces a significant economic benefit;

—accordingly, it is necessary to concentrate the efforts of scientists and organizations on the most promising aspects for unification of the different approaches to the subject and development, on this basis, of effective methods for synthesis of structures of automated management;

—it is necessary to pay much attention to the development and methodology of use of multi-purpose and problem-oriented software for synthesis of structures on the basis of a combination of optimization and simulation models.

In the opinion of the participants, the seminar was conducted on a high level, fostering a lively exchange of experience on the questions of synthesis of the structures of automated management in large-scale systems.

Intellectualization of the Processing of Natural Sciences Information

907G0134B Kiev UPRAVLYAYUSHCHIYE SISTEMY I MASHINY in Russian No 2, Feb 90, pp 124-126

[Article by P. V. Sigorskiy]

[Text] From 25 Sep to 2 October 1989, at Sevastopol, the first All-Union Seminar "Intellectualization of the Processing of Natural Sciences Information" was held, organized by the USSR Academy of Sciences, its Computer Center, the Academy's All-Union Research Institute of Systems Research, the Institute of Computer Science im. Glushkov of the Ukrainian SSR Academy of Sciences, and Simferopol State University. Taking part in the proceedings were around 80 scientists and specialists, representing more than 30 scientific-research and design organizations, technical colleges, industrial and scientific-creative associations, as well as members of the USSR Ministry of the Radio Industry and the Ministry of Instruments Building. The agenda of the seminar included 29 reports and lectures by prominent scientists in the field of information science, as well as discussions on the range of issues surveyed.

The sponsor of the seminar was the Scientific Research Institute of Consumer Electronics (Lvov), which during the time of the seminar organized a display of foremost projects in the field of optical disk drives for the PC, laser video and audio equipment, as well as high-quality sound amplification and acoustic equipment.

The seminar was opened by the chairman of the organizing committee, Academician V. S. Mikhalevich, who took up the urgency of the issue of intellectualization of information processing in the development of new information technologies. Today our society is entering a stage of all-encompassing computerization and "informatization," when modern computers are replacing many of the human productive functions—physical and mental. A mechanized "nervous system" of the economy is coming into being, just as the industrial "skeletal-muscular" system was once created. The technical foundation for these processes will be the latest means of gathering, transmitting, and processing information, primarily the computer, and the connecting link will be information science, i.e., a new technology of processing and making use of information in various areas of social practice.

The creation of various management, diagnostic, and simulation packages on the basis of modern computers has shown the decisive role of information and the whole cycle of its processing in social development. Work should not be conducted on an overly empirical level. And the decisive part here should be played by information science—the science which concentrates on the user aspects of computerization.

The working section of the seminar encompassed three main areas: problems of recognition, questions of design of artificial intelligence, and problems of information processing which occur during the use of laser optical disk drives.

Much interest on the part of the seminar participants was elicited by the lecture of K. V. Rudakov (Computer Center, USSR Academy of Sciences, Moscow) "Qualitative natural sciences information in problems of recognition," in which he set forth the basic ideas of algebraic correction of heuristics used in solving problems in areas that do not lend themselves to mathematical formalization. He demonstrated that the main tool in solving problems of such class are the universal constraints which define the particular problem. He also analyzed several classes of constraints which arise during the solving of extrapolation problems.

V. B. Britkov (All-Union Scientific Research Institute of Systems Research [VNIISI], USSR Academy of Sciences, Moscow), in his report "Intelligent information systems," talked about the basic approaches to the design of the intelligent systems being developed at the VNIISI.

Mathematical tools intended for description, economical coding, and processing of groups of images were examined in the report of M. I. Shlezinger (Institute of Computer Science, Ukrainian SSR Academy of Sciences, Kiev). These tools are two-dimensional grammars—a formalism resembling the familiar one-dimensional grammars, but taking into account the two-dimensional nature of the objects being generated. The proposed approach is based on specification of the groups of objects being recognized with the help of two-dimensional grammars, which is the reason for its strict orientation to the class of processing problems. The existence of this strict orientation is responsible for the most significant differences from other known approaches lacking such strict point of reference. The proposed formalism is an elaboration of the theory of recognition in a direction enabling formal specification of all a priori information about the class of objects being recognized so that the required function determining whether an object belongs to the class is distinctly obtained from this specification.

Aspects of image processing was also the subject of the report of G. L. Gimelfarb (Institute of Computer Science, Ukrainian SSR AS, Kiev), "Methods of statistical recognition in problems of low-level image processing," which examined issues of constructing effective models of signal segments and analysis of the image on the designated segments.

The report of S. V. Ablameyko (ITK [Institute of Technical Computer Science] Byelorussian SSR Academy of Sciences, Minsk) "Algorithms and programs for processing and recognition of graphic images" defined the basic stages of the process of formation of a digital model of a graphic image in terms of a raster representation, presented algorithms and programs for preliminary image processing, and presented an algorithm for distinguishing the contour of an object that is able to approximate the metrics of the object in tracking mode.

Among the reports and lectures devoted to algebraic programming, much interest was generated by the report of A. A. Letichevskiy (Institute of Computer Science, Ukrainian SSR AS, Kiev) "Algebraic programming in artificial intelligence problems," in which he set forth the general principles of design of software on the basis of algebraic and logic resources. The speaker took up the main features of the experimental system of algebraic programming, APS-1, developed under his leadership.

V. I. Donskoy (Simferopol State University), in his report "Dual expert systems and methods of making optimal decisions," analyzed the algorithmic possibilities of the optimization criterion of the decision making process as part of a decision making support system and demonstrated the comparative ease with which complexity and indeterminacy may be factored into integrated expert systems when the appropriate intelligent program packages are available.

Questions of the use of modern mathematical methods in the design of consumer electronics and peripherals for the PC were the subject of the report of V. V. Sekretaryuk (Scientific Research Institute of Consumer Electronics, Lvov), "Problems of processing of image-containing information during the development of electronic appliances." Methods of adaptive filtration of the television signal are used in the development of analog-digital and digital television sets. If we add to this the implementation of various television effects, a rather important problem of image processing arises, for the solving of which the Institute is making successful use of pattern recognition methods.

The report of L. V. Varichenko (Scientific Research Institute of Consumer Electronics, Lvov) "The method of image texture analysis" was devoted to a description of a method of hardware implementation of image texture analysis. The primary operation employed is computation of the spectral power density, which may serve as a characteristic that is analogous to the autocorrelation function, widely used in the familiar texture analysis algorithms.

L. M. Shashuk (Research Institute of Consumer Electronics, Lvov) in his lecture "On the status and prospects for use of laser optical disks in applied systems" took up the problems arising in the development and use of optical disk drives and ways of overcoming them. This material aroused much interest in the specialists occupied in design and development of intelligent systems in which it is necessary to process large volumes of information.

There was ample presentation of the issues of constructing actual intelligent information processing systems. There were the reports of V. V. Shafranskiy (Computer Center, USSR Academy of Sciences, Moscow) "Intelligent systems in the planning and design of production," V. V. Yurchenko (VNIISI, USSR Academy of Sciences, Moscow) "Representation of knowledge by means of functional networks," and Z. B. Bosikashvili (GPI [Georgian Polytechnic Institute], Tbilisi) "An expert system for automated gathering and accumulation of information."

In the discussions on the materials of the individual reports, and also in the general discussions on the working areas of the seminar, the following main development trends were identified for the means and methods of intellectualization of the processing of information in the natural sciences:

—expansion of the research in the field of pattern recognition;

—development of new mathematical methods for image analysis and processing;

—development of work in the creation of expert systems;

—ensuring reliable research findings;

—extensive introduction of new generations of PCs and peripherals.

The seminar fostered an exchange of opinions and ideas, created a setting for working out valid approaches to the problems of synthesis of intelligent systems, encouraged the formulation of the most important and promising problems in this field, and strengthened the creative and professional contacts among the organizations and individual specialists working in the field of artificial intelligence and related disciplines.

A clear tendency was evident to move away from purely theoretical constructs toward extensive introduction of the ideas of pattern recognition theory in the practice of scientific research, industry, and consumer electronics.

The modern day PCs and peripherals, the image recognition and processing systems, the expert systems, the systems for nondestructive inspection and technical diagnostics—these are some of the results actually achieved today on the basis of introducing the ideas and methods of the theory of pattern recognition and set forth at the seminar.

The seminar participants expressed their gratitude to the sponsor, the Scientific Research Institute of Consumer Electronics (Lvov). A resolution was adopted to hold the second seminar in Jan 1991 in the Lvov oblast.

END OF

FICHE

DATE FILMED

15 Aug. 1990